Trapped Ion Magnetic Resonance:

Concepts and Designs

Thesis by

Pedro José Pizarro

In Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy

California Institute of Technology
Pasadena, California

1994

(Submitted September 29, 1993)

Acknowledgements

There are many people whose input, support, and friendship enriched my years at Caltech. Prof. Dan Weitekamp introduced me to an exciting branch of chemical physics, helped me to contribute to the invention of promising techniques, and opened professional doors that I cannot take for granted. I especially enjoyed interacting with Dan and his family on a personal level; I know of few advisors who would have been equally excited to ski Mt. Baldy in the midst of freezing rain!

The members of the Weitekamp group enhanced my graduate experience in very special ways. The scientific talent and breadth of every group member contributed greatly to the evolution of my dissertation work. I am especially grateful for my collaborations with David Shykind, Len Mueller, and Gary Leskowitz. I will remember fondly the time spent with the group members and their families outside the lab. Thanks go to all for their friendship: Dan Jones, Narayanan (K.D.) Kurur, Jack Hwang (thanks for the skiing addiction!), Len and Shelley Mueller, Steve and Laura Buratto, Dave Shykind and Maria Giorgi, Herman Cho, Margat and Charlie Werner, Paul Carson, John Marohn, Mike and Arezoo Miller, and Gary Leskowitz.

Thanks go to the other inhabitants of the Noyes subbasement for their friendship. The members of the Zewail and Kuppermann groups made life below the sewer line enjoyable; special thanks go to Earl Potter, Dean Willberg, Jennifer Herek, and Carrie Stroud. Thanks also to Mike Rock, Marc Hillmyer, Laura Thomas, and Todd and Missy Richmond.

Prof. Jack Beauchamp and the members of his group, especially Karl Irikura, deserve special thanks for teaching me about experimental ion cyclotron resonance. Tom

Boyce and Valerie Okinaka of the Campus Computation Organization provided much appreciated computer assistance.

The members of the Residence Life staff added a new dimension to my life for three years. Many thanks go to my fellow Resident Associates, and especially to Kim West for her superb job as Director and, more importantly, her friendship. I also thank those residents of Page House and other Caltech houses who made the sometimes difficult job of the R.A.'s more enjoyable.

Deep gratitude and love go to my family. My parents, Pedro and Eva Pizarro, deserve my eternal thanks for the many sacrifices through which they provided me with an excellent education. Their love, kindness, and faith in God will always be with me. My sister, Evita, has added light-hearted balance to my life and has grown into the best of adult friends.

I thank my wife, Monica Kohler, for our life together. We have shared many different experiences: graduate student life, joint employment and crisis handling as Resident Associates, and, lest this sound too serious, a lot of fun! I have also enjoyed spending time with her parents, Frank and Gertrud Kohler, as well as her sister Andrea and sister and brother-in-law Christine and Robert Koehler. My love and deepest gratitude go to Monica for being a loving spouse, a fun companion, and my best friend.

Abstract

A novel spectroscopy of trapped ions is proposed which will bring single-ion detection sensitivity to the observation of magnetic resonance spectra. The approaches developed here are aimed at resolving one of the fundamental problems of molecular spectroscopy, the apparent incompatibility in existing techniques between high information content (and therefore good species discrimination) and high sensitivity. Methods for studying both electron spin resonance (ESR) and nuclear magnetic resonance (NMR) are designed. They assume established methods for trapping ions in high magnetic field and observing the trapping frequencies with high resolution (<1 Hz) and sensitivity (single ion) by electrical means. The introduction of a magnetic bottle field gradient couples the spin and spatial motions together and leads to a small spin-dependent force on the ion, which has been exploited by Dehmelt to observe directly the perturbation of the ground-state electron's axial frequency by its spin magnetic moment.

A series of fundamental innovations is described in order to extend magnetic resonance to the higher masses of molecular ions (100 amu $\cong 2\times10^5$ electron masses) and smaller magnetic moments (nuclear moments $\cong 10^{-3}$ of the electron moment). First, it is demonstrated how time-domain trapping frequency observations before and after magnetic resonance can be used to make cooling of the particle to its ground state unnecessary. Second, adiabatic cycling of the magnetic bottle off between detection periods is shown to be practical and to allow high-resolution magnetic resonance to be encoded pointwise as the presence or absence of trapping frequency shifts. Third, methods of inducing spin-dependent work on the ion orbits with magnetic field gradients and Larmor frequency irradiation are proposed which greatly amplify the attainable shifts in trapping frequency.

The dissertation explores the basic concepts behind ion trapping, adopting a variety of classical, semiclassical, numerical, and quantum mechanical approaches to derive spin-dependent effects, design experimental sequences, and corroborate results from one approach with those from another. The first proposal presented builds on Dehmelt's experiment by combining a "before and after" detection sequence with novel signal processing to reveal ESR spectra. A more powerful technique for ESR is then designed which uses axially synchronized spin transitions to perform spin-dependent work in the presence of a magnetic bottle, which also converts axial amplitude changes into cyclotron frequency shifts. A third use of the magnetic bottle is to selectively trap ions with small initial kinetic energy. A dechirping algorithm corrects for undesired frequency shifts associated with damping by the measurement process.

The most general approach presented is *s*pin-*l*ocked *i*nternally *r*esonant *i*on *c*yclotron *e*xcitation, a true continuous Stern-Gerlach effect. A magnetic field gradient modulated at both the Larmor and cyclotron frequencies is devised which leads to cyclotron acceleration proportional to the transverse magnetic moment of a coherent state of the particle and radiation field. A preferred method of using this to observe NMR as an axial frequency shift is described in detail. In the course of this derivation, a new quantum mechanical description of ion cyclotron resonance is presented which is easily combined with spin degrees of freedom to provide a full description of the proposals.

Practical, technical, and experimental issues surrounding the feasibility of the proposals are addressed throughout the dissertation. Numerical ion trajectory simulations and analytical models are used to predict the effectiveness of the new designs as well as their sensitivity and resolution. These checks on the methods proposed provide convincing evidence of their promise in extending the wealth of magnetic resonance information to the study of collisionless ions via single-ion spectroscopy.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

The study of ions by cyclotron resonance techniques continues to expand more than sixty years after its introduction.[1] The many types of cyclotron experiments all rely on the determination of ion mass from its inverse relationship to the measured cyclotron frequency in a given magnetic field. Earlier ion cyclotron resonance (ICR) experiments[2,3] scanned through a single frequency at a time and were, thus, relatively time consuming. The advent of more sensitive detection electronics and of fast Fourier transformation by computers led to Fourier transform ion cyclotron resonance (FT/ICR) experiments,[4,5] which obtain the entire frequency spectrum at once. Chemical interest in ICR goes well beyond mass spectrometry. The ion selectivity afforded by double resonance techniques, where certain ions are purged from the cell by ICR excitation while other ions are monitored, is useful in observing reaction rates.[6,7] All ICR experiments rely on the ion trapping due to cyclotron motion. The magnetic field confines ions transversely in cyclotron orbits, and the addition of electrostatic plates leads to harmonic axial motion. This combination of electric and magnetic fields keeps ions trapped as long as several seconds in typical chemical applications. To study single trapped ions, however, physicists employ precise field configurations such as the Penning trap:[8] a single electron has been trapped for as long as ten months.[9] Values of the Lande g factor measured with a single electron in a Penning trap are the most accurate to date.[10]

While ICR and related techniques are highly sensitive, the only observable they measure is mass (with the exception of the single electron g experiments). Conventional ICR thus cannot distinguish two structurally different species of equal mass. This is in marked contrast to nuclear magnetic resonance (NMR) and electron spin resonance

(ESR), which yield a wealth of information from low-energy spectral features, but have poor sensitivity. This is a basic problem of experimental chemical physics and extends to other forms of spectroscopy: discrimination and sensitivity seem incompatible. This dissertation presents the theoretical framework for a spectroscopy of ions which will use the fine sensitivity of trapped-ion techniques to observe highly discriminating magnetic resonance spectra. The case of ions with spin 1/2 magnetic moments is treated here for concreteness and relative theoretical ease, but the results may be extended to more complex magnetic characteristics and to the rotational spectroscopy of ions.

1.1 Background

The cyclotron was first introduced by Lawrence and Livingston in 1932 as a tool for "the production of high speed light ions without the use of high voltages."[1] Four years later, Penning used magnetic and electric fields to increase the lifetime of electrons within a discharge.[8] These experiments provided a solid base on which the field of ion cyclotron resonance spectroscopy was later built. Its importance in studying chemical reactions was demonstrated in the late 1960's by Beauchamp, Baldeschwieler, and coworkers,[11,12] using double resonance techniques[6] where changes in the concentration of a product ion (monitored via the intensity of its cyclotron signal) were monitored as a function of the energy of a reactant ion (varied by resonantly pumping its cyclotron energy). The manipulation of ion populations in an ion trap was extended by the introduction of resonant ion ejection as a tool for isolating interesting chemical species.[7] The advent of Fourier transform ICR in 1974 brought increased speed and sensitivity to these applications.[4,5,13]

While ICR development in chemistry labs from the late 1960's on had as one of its fundamental goals increased bandwidth in order to study ions of widely ranging masses

simultaneously, research in physics labs at the time seemed directed orthogonally: narrow-bandwidth detection with single-ion sensitivity to measure ionic masses with unprecedented precision. The tools developed would thus enable the metrology of fundamental constants. The first milestone on this road was the trapping and detection of a single electron in a 6 V deep Penning trap by Wineland, Ekstrom, and Dehmelt in 1973;[14] the electron was viewed as bound to the earth (through the Penning trap), rather than to an atom, and was thus termed "*geo*nium." This experiment, in turn, led to the metrology of the electron spin (Lande) g factor by measuring the spin dependence of the detected axial frequency in the presence of a weak magnetic bottle.[15,16,17,18] This measurement of the electron magnetic moment from geonium spectra, which earned Dehmelt the Nobel Prize in physics, provided a dual motivation to develop the proposals included in this dissertation: its success suggested that magnetic resonance might be carried out on trapped atomic and molecular ions, while its technical feasibility only for masses as light as the electron necessitated the search for greatly amplified spin-dependent effects which culminated in schemes for extended periods of spin-dependent *work* on trapped ion motions.[19,20,21]

More recent developments in trapped ion physics have extended the detection sensitivity to single atomic and molecular ions by using one of two techniques: optical detection or electrical detection. Optical methods use laser cooling to minimize ion motions and therefore the effects of field imperfections, increasing resolution; cyclotron frequency measurements accurate to up to 1 part in 10^{13} could be possible in the future.[22] The fluorescence signal from single ions with appropriate internal level structure can then be monitored[23,24] via schemes based on Dehmelt's electron shelving.[25] Optical detection is not generally applicable, since it is limited to ions which can be made to fluoresce. Electrical detection has the advantage of promising single-ion sensitivity for any ion given a large enough orbit, at the expense of the relative ease with which high resolution is available after laser cooling to the single quantum level. In electrical detection, the ion's

image current induced on the trap electrodes couples the ion to electrical circuits attached to the trap. Pritchard and coworkers have pioneered single-ion sensitivity in molecular ion detection, observing the axial motion of single trapped ions by using a high-Q superconducting tank circuit to couple the ion trap to an rf SQUID.[26] Single-ion cyclotron resonance (SICR) measurements in a Penning trap have already been carried out with mass accuracy $\Delta M/M = 4\times10^{-10}$.[27] Simultaneous observation of two ions in a Penning trap with single-ion sensitivity,[28] as well as classical squeezing with parametric drives,[29] promises to extend the accuracy to a part in 10^{12}. The magnetic resonance methods proposed presented in this dissertation will employ electrical detection with single-ion sensitivity and moderate resolution requirements.

1.2 Outline

The classical description of ions stored in a Penning trap is presented in Chapter 2. The effects of superimposed field gradients are considered, with particular attention paid to the case of the so-called magnetic bottle field gradient,[10,30] which couples spin to the spatial degrees of freedom. Trajectory simulations via numerical integration of the classical equations of motion are introduced as a powerful tool for characterizing and designing ion trap experiments. Ion cyclotron resonance is described within the classical formalism of this chapter.

Chapter 3 presents a quantum mechanical description of ion trapping which replaces the Penning trap's normal modes with corresponding harmonic oscillators. Position and momentum operators are thus replaced with quantum raising and lowering operators. A strength of this quantum formalism is the relative ease with which it allows the effect of field gradients to be considered through perturbation theory. This technique is illustrated for the case of the magnetic bottle. The quantum operators are used to describe classical systems by constructing semiclassical wavepackets which describe ion

motion. This treatment is used in a rotating frame calculation to derive cyclotron radius changes due to ICR.

The experiments proposed in Chapter 4, *i*on *o*rbit *n*udging *t*hrough *r*esonance *i*nduce *c*yclotron *k*inematics (IONTRICK), study a paramagnetic (spin 1/2) ion confined in a Penning trap with a magnetic bottle superimposed, coupling spin and space operators. This allows ESR magnetic transitions to be observed by monitoring changes in spatial quantities, in particular the ion cyclotron frequency. A proposal to monitor the cyclotron frequency change due to a change in axial energy after an induced spin transition is presented. This idea is then extended by the use of multiple π pulses synchronized to the axial motion such that spin-dependent work can be done on the ion's axial motion and monitored through the cyclotron frequency. Results from simulations establish the feasibility of the IONTRICK proposal, and its technical requirements and challenges are discussed.

Chapter 5 presents a more general proposal, *s*pin-*l*ocked *i*nternally *r*esonant *i*on *c*yclotron *e*xcitation (SLIRICE), enabling the observation of NMR transitions via electrical ion detection. A historical precedent which provided some of the motivation for SLIRICE, Bloom's deflection of neutral molecular beams by radiofrequency field gradients (the "transverse Stern-Gerlach effect"),[31,32] is discussed. The conditions under which a magnetic field gradient modulated at both the Larmor and cyclotron frequencies leads to cyclotron acceleration proportional to the transverse magnetic moment of a coherent state of the particle and radiation field are derived both semiclassically and quantum-mechanically. The need for two additional magnetic field gradients is clarified: a radiofrequency spin-lock field preserves spin coherence, and a static magnetic bottle translates the spin-dependent cyclotron work into a corresponding shift in the axial trapping frequency. Numerical simulations are employed to support the validity of the SLIRICE proposal.

Concluding remarks are presented in Chapter 6. This is followed by two Appendices. Appendix A describes the programs used to integrate the equations of ion motion numerically. Appendix B describes the programs which simulate experimental timelines analytically and allow the evaluation of expected signal-to-noise (S/N) ratios. In each appendix, sample program source codes and run-time instructions are presented.

1.3 References

[1]Lawrence, E.O., and M.S. Livingston, Phys. Rev. **40**, 19 (1932).

[2]Beauchamp, J.L., Ann. Rev. Phys. Chem. **22**, 527 (1971).

[3]Sharp, T.E., J.R. Eyler, and E. Li, Int. J. Mass Spectrom. Ion Phys. **9,** 421 (1972).

[4]Marshall, A.G., Acc. Chem. Res. **18**, 316 (1985).

[5]Comisarow, M.B., Anal. Chim. Acta **178**, 1 (1985).

[6]Anders, L.R., J.L. Beauchamp, R.C. Dunbar, and J.D. Baldeschwieler, J. Chem. Phys. **45**, 1062 (1966).

[7]Beauchamp, J.L., and J.T. Armstrong, Rev. Sci. Inst. **40**, 123 (1969).

[8]Penning, F.M., Physica (Utrecht) **3**, 873 (1936).

[9]Gabrielse, G., H. Dehmelt, and W. Kells, Phys. Rev. Lett. **54**, 537 (1985).

[10]Brown, L.S., and G.Gabrielse, Rev. Mod. Phys. **58**, 233 (1986).

[11]Beauchamp, J.L., L.R. Anders, and J.D. Baldeschwieler, J. Am. Chem. Soc. **89**, 4569 (1967).

[12]Baldeschwieler, J.D., Science **159**, 263 (1968).

[13]Comisarow, M.B., and A.G. Marshall, Chem. Phys. Lett. **25**, 282 (1974).

[14]Wineland, D., P. Ekstrom, and H. Dehmelt, Phys. Rev. Lett. **31**, 1279 (1973).

[15]Van Dyck, R., Jr., P. Ekstrom, and H. Dehmelt, Nature **262**, 776 (1976).

[16]Van Dyck, R.S., Jr., P.B. Schwinberg, and H.G. Dehmelt, Phys. Rev. Lett. **38**, 310 (1977).

[17]Van Dyck, R.S., Jr., P.B. Schwinberg, and H.G. Dehmelt, in *New Frontiers in High Energy Physics*, edited by B. Kursunoglu, A. Perlmutter, and L. Scott (Plenum, New York, 1978).

[18]Van Dyck, R.S., Jr., P.B. Schwinberg, and H.G. Dehmelt, in Atomic Physics **9**, edited by R.S. Van Dyck, Jr. and E.N. Fortson (World Scientific, Singapore, 1984).

[19]Bowers, C.R., S.K. Buratto, P.J. Carson, H.M. Cho, J.Y. Hwang, L. J. Mueller, P.J. Pizarro, D.N. Shykind, and D.P. Weitekamp, SPIE Proc. **1435**,36 (1991).

[20]Weitekamp, D.P., and P.J. Pizarro, U.S. Patent No. 4,982,088 (1991).

[21]Pizarro, P.J., and D.P. Weitekamp, Bull. Mag. Reson. **14**, 220 (1992).

[22]Wineland, D.J., J.J. Bollinger, and W.M. Itano, Phys. Rev. Lett. **50**, 628 (1983).

[23]Wineland, D.J. ,W.M. Itano, J.C. Bergquist, and R.G. Hulet, Phys. Rev. A **36**, 2220 (1987).

[24]Diedrich, F., J.C. Bergquist, W.M. Itano, and D.J. Wineland, Phys. Rev. Lett. **62**, 403 (1989).

[25]Dehmelt, H., Bull. Am. Phys. Soc. **20**, 60 (1975).

[26]Weisskoff, R.M., G.P. Lafyatis, K.R. Boyce, E.A. Cornell, R.W. Flanagan, Jr., and D.E. Pritchard, J. Appl. Phys. **63**, 4599 (1988).

[27]Cornell, E.A., R.M. Weisskoff, K.R. Boyce, R.W. Flanagan, Jr., G.P. Lafyatis, and D.E. Pritchard, Phys. Rev. Lett. **63**, 1674 (1989).

[28]Cornell, E.A., K.R. Boyce, D.L.K. Fygenson, and D.E. Pritchard, Phys. Rev. A **45**, 3049 (1992).

[29]DiFilippo, F., V. Natarajan, K.R. Boyce, and D.E. Pritchard, Phys. Rev. Lett. **68**, 2859 (1992).

[30]Gabrielse, G., and H. Dehmelt, Phys. Rev. Lett. **55**, 67 (1985).

[31]Bloom, M., and K. Erdman, Can. J. Phys. **40**, 179 (1962).

[32]Bloom, M., E. Enga, and H. Lew, Can. J. Phys. **45**, 1481 (1967).

Chapter 2: Ion Trapping and its Classical Description

The classical equations of ion motion in a Penning trap are presented in this chapter. The analytical solution of these equations will be shown to yield cyclotron, axial, and magnetron motional modes with respective characteristic frequencies. The effect of field gradients on these modes will be addressed, with particular attention paid to magnetic bottle field gradients. Ion trajectory simulations via numerical integration of the equations of motion will be presented as a valuable tool to elucidate ion motion when field gradients prevent the existence of analytical solutions. The chapter concludes with a discussion of ion cyclotron resonance, with particular attention to the derivation of cyclotron excitation and to experimental details.

2.1 Equations of motion

The general equation of motion of a charged particle of mass m in an electromagnetic field (ignoring spin) is[1] (using the mks system of units)

$$\frac{d^2\bar{r}}{dt^2} = \frac{q}{m}\left(\vec{E} + \bar{v} \times \vec{B}\right) \qquad [2.1]$$

where $\bar{r}$ represents the ion coordinates, t is time, q is the ionic charge, $\vec{E}$ is the electric field, $\bar{v}$ is the ion velocity, and $\vec{B}$ is the magnetic field. In the case of zero electric field and a homogeneous, time-independent magnetic field $\vec{B} = B_0\hat{z}$, ion motion is unrestricted along the field and confined to a circular oscillation perpendicular to the field at the cyclotron frequency,

$$\omega_c = \frac{qB_0}{m}. \qquad [2.2]$$

The direction of rotation is clockwise for a positively charged ion (looking along the $-\hat{z}$ direction). The cyclotron motion thus constitutes a two-dimensional trap. To complete the trapping in all three dimensions, a steady electric field is applied along the $\hat{z}$ direction by placing flat plates of potential V_0 perpendicular to the magnetic field (at $\pm z_0$), such that $qV_0 > 0$. The ion will then be kept confined to $-z_0 \le z \le z_0$ as long as its axial energy is not greater than qV_0. Conventional ICR cells utilize such flat plates to make cubic or rectangular cells. Since the electric field near the center of these cells is nearly quadrupolar, the axial motion is harmonic.[2] Trapping times on the order of seconds can be achieved with these relatively simple cells.

The Penning trap[3,4,5] uses three hyperbolic electrodes (two endcaps and a central ring), as shown in Fig. 2.1, to produce a more exactly quadrupolar electric field; small deviations can be corrected with compensation electrodes. The two endcaps are along the two branches of the hyperbola of revolution

$$z^2 = z_0^2 + \frac{\rho^2}{2}, \qquad [2.3a]$$

and the ring electrode lies along the hyperbola of revolution

$$z^2 = \frac{\rho^2 - \rho_0^2}{2}. \qquad [2.3b]$$

The coordinate ρ is the radial distance of the ion from the **z** axis; that is,

$$\rho^2 = x^2 + y^2. \qquad [2.4]$$

Defining a characteristic trap dimension d by[4]

$$d^2 = \frac{1}{2}\left(z_0^2 + \frac{\rho_0^2}{2} \right), \qquad [2.5]$$

the electric field now takes the form

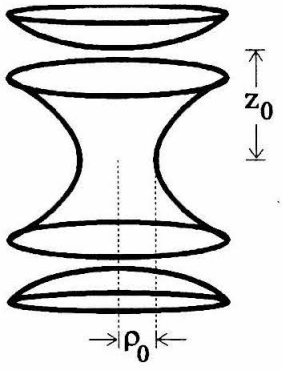


Fig. 2.1. Scale drawing of the electrodes of the Penning trap used in the single electron experiments. Adapted from Ref. 4.

$$\vec{E} = \frac{V_0}{d^2}\left(\frac{x}{2}\hat{x} + \frac{y}{2}\hat{y} - z\hat{z}\right). \qquad [2.6]$$

The equations of motion are

$$\begin{pmatrix} \frac{d^2x}{dt^2} \\ \frac{d^2y}{dt^2} \\ \frac{d^2z}{dt^2} \end{pmatrix} = \begin{pmatrix} \frac{\omega_z^2}{2}x + \omega_c\frac{dy}{dt} \\ \frac{\omega_z^2}{2}y - \omega_c\frac{dx}{dt} \\ -\omega_z^2 z \end{pmatrix}, \qquad [2.7]$$

with ω_z defined as the angular frequency of the purely harmonic axial motion,

$$\omega_z^2 = \frac{qV_0}{md^2}. \qquad [2.8]$$

The equations are solved analytically by introducing two new transverse components of motion[4] or by introducing trial solutions for x and y that are superpositions of harmonic oscillations at two different frequencies. These give equivalent descriptions of the behavior, namely, that the transverse motion is the superposition of a fast cyclotron motion (at a modified cyclotron frequency, ω_+) and a slower magnetron motion (at the magnetron frequency, ω_-). The relevant frequencies are

$$\omega_{\pm} = \frac{1}{2}\left(\omega_c \pm \sqrt{\omega_c^2 - 2\omega_z^2}\right). \qquad [2.9]$$

Introducing the transverse vectors $\vec{V}^{(\pm)}$ such that

$$\vec{V}^{(\pm)} = \frac{d\vec{\rho}}{dt} + \text{sign}(q)[\omega_{\mp}\hat{z} \times \vec{\rho}], \qquad [2.10]$$

where the dependence on the sign of the ionic charge has been made explicit by sign(q), which is "+" for $q > 0$ and "−" for $q < 0$, the transverse Hamiltonian of the system can be expressed as[4]

$$H_{\rho} = \frac{m}{2}\frac{\omega_+ \vec{V}^{(+)^2} - \omega_- \vec{V}^{(-)^2}}{\omega_+ - \omega_-}. \qquad [2.11]$$

The cyclotron and axial motions are bound harmonic oscillations. Although harmonic, the magnetron motion is motion around a potential hill, not bound in a potential well, and gives a negative contribution to the total energy. The relative magnitudes of the frequencies are

$$\omega_+ >> \omega_z >> \omega_-. \qquad [2.12]$$

This hierarchy is typically most pronounced for smaller particles. Considering a magnetic field of 1 T, with $V_0 = 10$ V and $d = 1$ cm, the respective frequencies ν_+, ν_z, and ν_- (with $\omega_i = 2\pi\nu_i$) are 28 GHz, 21 MHz, and 8.0 kHz for an electron, 15 MHz, 490 kHz, and 8.0 kHz for a proton, and 145 kHz, 49 kHz, and 8.4 kHz for a 100 amu ion.

2.2 Modification of trapped ion motion by a magnetic bottle

In order to incorporate spin dependence into trapped ion experiments, it will be necessary to couple spin to the spatial ion modes. This can be achieved by devising situations analogous to the Stern-Gerlach experiment, which first established the quantization of spin angular momentum.[6,7,8] In the Stern-Gerlach apparatus, an inhomogeneous magnetic field coupled the spin of the valence 5*s* electron of silver atoms

to the motion of the atoms in a beam, splitting the beam into two branches, one for each of the quantized spin 1/2 orientations. A trapped ion analogy is the change in frequency of the Penning trap modes in proportion to spin magnetic moment.[9,10] The symmetry of this problem suggests the use of a cylindrically symmetric gradient. The simplest such field is a magnetic bottle of the form[4]

$$\Delta\vec{B} = B_2\left[\left(z^2 - \frac{\rho^2}{2}\right)\hat{z} - z\vec{\rho}\right], \qquad [2.13]$$

where

$$\vec{\rho} = x\hat{x} + y\hat{y}. \qquad [2.14]$$

In the mks system, B_2 has units of T/m^2, which are equivalent to G/cm^2. The reason for the term "bottle" is obvious when a magnetic moment μ is aligned along and on the **z** axis. Then, the correction to the Penning trap Hamiltonian is

$$\Delta H(\rho = 0, z) = -\mu B_2 z^2, \qquad [2.15]$$

and, for $\mu B_2 < 0$, an axial harmonic potential traps the moment along the **z** axis.[4] Fields similar to the magnetic bottle arise as inhomogeneities in laboratory magnets; the "z^2" shim coil of an NMR magnet has approximately this form.

Frequency shifts due to such inhomogeneities can be analyzed within the framework of classical mechanics. The classical equations of motion for an ion in a Penning trap are modified by the presence of a magnetic bottle. The extra Lorentz force due to the bottle must thus be added to the equations of motion [2.7]. The addition of an inhomogeneous field also introduces a spin-dependent force, $\vec{F}_s$, given by[11]

$$\vec{F}_s = \vec{\nabla}(\vec{\mu} \bullet \vec{B}) \cong (\vec{\mu} \bullet \vec{\nabla})\vec{B}. \qquad [2.16]$$

For classical trajectories in the absence of Larmor frequency magnetic fields, the quantization direction for the spin magnetic moment adiabatically follows the

instantaneous direction of the total magnetic field, $\vec{B}$, and the direction of $\vec{\mu}$ is parallel to the direction of $\vec{B}$, that is,

$$\hat{\mu} = \hat{B} = \frac{\vec{B}}{|\vec{B}|} \qquad [2.17]$$

(where $\vec{\mu} = \mu\hat{\mu}$, and μ carries the sign of $\vec{\mu}$).

The spin force due to the total magnetic field in this case is

$$\vec{F}_s = \frac{\mu B_2}{|\vec{B}|}\left[x\left(-B_0 + \frac{B_2}{2}\rho^2 \right)\hat{x} + y\left(-B_0 + \frac{B_2}{2}\rho^2 \right)\hat{y} + 2z\left(B_0 + B_2 z^2 \right)\hat{z} \right]. \qquad [2.18]$$

In this semiclassical treatment, this spin-dependent (and, therefore, intrinsically quantum mechanical) force is added to the classical equations of motion [2.7], which now become

$$\begin{pmatrix} \frac{d^2x}{dt^2} \\ \frac{d^2y}{dt^2} \\ \frac{d^2z}{dt^2} \end{pmatrix} = \begin{pmatrix} \omega_z^2 x + \left[\omega_c + \frac{qB_2}{m}\left(z^2 - \frac{\rho^2}{2} \right) \right]\frac{dy}{dt} + \frac{qB_2}{m} yz\frac{dz}{dt} + \frac{\mu B_2}{m|\vec{B}|}\left(-B_0 + \frac{B_2}{2}\rho^2 \right)x \\ \omega_z^2 y - \left[\omega_c + \frac{qB_2}{m}\left(z^2 - \frac{\rho^2}{2} \right) \right]\frac{dx}{dt} - \frac{qB_2}{m} xz\frac{dz}{dt} + \frac{\mu B_2}{m|\vec{B}|}\left(-B_0 + \frac{B_2}{2}\rho^2 \right)y \\ -\omega_z^2 z + \frac{qB_2}{m} z\left(x\frac{dy}{dt} - y\frac{dx}{dt} \right) + \frac{2\mu B_2}{m|\vec{B}|}\left(B_0 + B_2^2 z^2 \right)z \end{pmatrix}. \qquad [2.19]$$

Due to the extensive coupling of these differential equations, the classical problem has no analytical solution, and may only be solved approximately or numerically; a numerical integration is discussed below. Quantum mechanical perturbation theory can also provide estimates of trapped ion behavior in a weak magnetic bottle, as will be shown in Chapter 3. Laukien[12] calculated the first order frequency shifts due to generic magnetic field gradients. Applying his results to the magnetic bottle (in the limit of axial and magnetron amplitudes small compared to the cyclotron radius ρ_c) leads to the cyclotron frequency shift

$$\Delta\omega_+ = -\frac{q}{2m} B_2 \rho_c^2 \qquad [2.20]$$

and the axial frequency shift

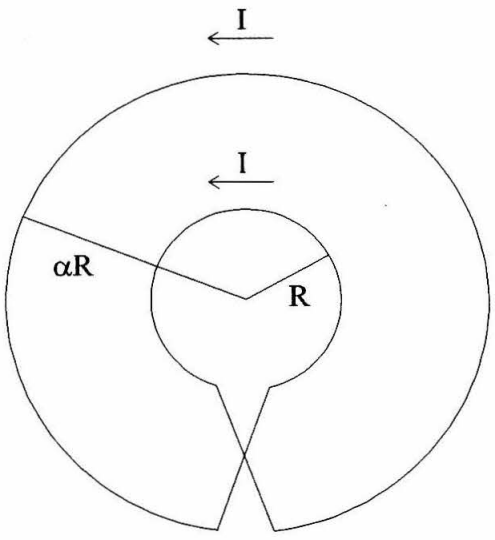


Fig. 2.2. Double loop variable magnetic bottle. Adapted from Ref. 5.

$$\Delta\omega_z = \frac{q\omega_c}{2m\omega_z} B_2 \rho_c^2. \qquad [2.21]$$

2.3 The variable magnetic bottle

The design of magnetic bottles is reviewed in detail by Brown and Gabrielse.[4] The bottle field is typically produced by a loop of ferromagnetic material in the **x-y** plane of the Penning trap, at the center of the **z** axis, and is constant in time. Variable bottles, also feasible, are a central component of the experiments proposed throughout this dissertation.

2.3.a Coplanar double loop variable magnetic bottle (superconducting flux transformer)

Van Dyck *et al.*[5] discuss the design and successful implementation of a variable bottle made from a single continuous superconducting NbTi loop twisted into two concentric loops (Fig. 2.2). The double loop is inserted into the homogeneous magnetic

field above its critical temperature; as it cools to the superconducting state, the enclosed flux is frozen in. Application of the field from a third magnet (an external solenoid of variable current) leads the now superconducting double loop to resist the new flux via Lenz's law, producing a persistent current. It is this current that adds a bottle field; since the field is proportional to the expelled flux, the apparatus is called a *superconducting flux transformer*. A single loop would have the same effect, but the double loop has an added advantage. There exists an optimum ratio, α, of the outer loop radius to the inner loop radius for which the only contribution to the field is the magnetic bottle itself. No other terms are added to the magnetic field, most importantly no zeroth-order term; in fact, a double loop with ratio α improves the uniformity of the homogeneous field at the center of the trap.[5] Thus, the double loop can serve as a shim to cancel out precisely any stray magnetic bottle-like inhomogeneities due to the trap itself (by creating a bottle field of opposite sign) and increase the uniformity of the remaining field. In the experiments proposed below, the main attraction of the double loop bottle is its variability and its capacity to produce large bottle fields, at least on the order of 100 to 1000 G/cm^2.

2.3.b Coaxial triple loop variable magnetic bottle (room temperature)

While the superconducting flux transformer succeeds in producing a variable magnetic bottle field, its requirement that the loops be immersed in liquid helium introduces a significant technical challenge. Even if this is done, the requirement for an external solenoid to drive the bottle complicates the geometry. As an alternative, a variable magnetic bottle may be designed from three coaxial current loops. Each loop is a simple non-persistent electromagnet, which could be at any convenient temperature. The end loops both have the radius R_1, and each is separated from the central loop (of radius R_0) by the distance d, as shown in Fig. 2.3. Note that a configuration with $R_0 > R_1$ will be ideally suited to use with a Penning trap, since the trap will fit nicely inside the loop assembly. The problem to be solved now is finding a choice of R_0, R_1, d, and the currents

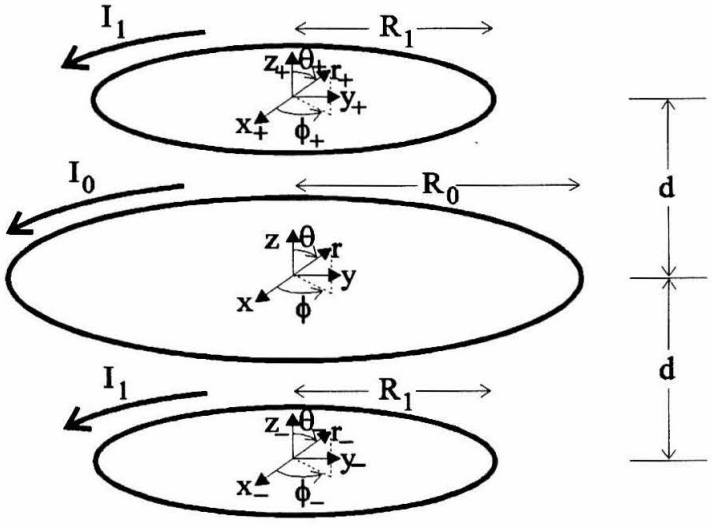


Fig. 2.3 Triple loop variable magnetic bottle. The individual Cartesian and spherical coordinate systems for each of the three loops are labeled: $(x_+,y_+,z_+)\leftrightarrow(r_+,\theta_+,\phi_+)$ for the top loop, $(x,y,z)\leftrightarrow(r,\theta,\phi)$ for the central loop, and $(x_-,y_-,z_-)\leftrightarrow(r_-,\theta_-,\phi_-)$ for the top loop.

I_1 (end loops) and I_0 (central loop) that will produce a pure magnetic bottle field, no homogeneous field, and minimum error terms.

The problem is solved by considering the vector potential for the system. Jackson[13] gives the vector potential for a single loop of radius R with current I, in spherical coordinates:

$$A_\phi(r,\theta)=\frac{4IR}{c\sqrt{R^2+r^2+2Rr\sin\theta}}\left[\frac{\left(2-k^2\right)K(k)-2E(k)}{k^2}\right], \qquad [2.22a]$$

$$A_z=A_r=0, \qquad [2.22b]$$

where

$$k^2 = \frac{rRr\sin\theta}{R^2 + r^2 + 2Rr\sin\theta}, \qquad [2.22c]$$

and K and E are the complete elliptic integrals. The total vector potential for the case of the triple loop assembly is the superposition of the respective single loop potentials. Working in the central loop coordinate system, the problem is solved by transforming the end loop potentials to this coordinate system through the use of the identities

$$r_+^2 = r^2 + d^2 - 2rd\cos\theta, \qquad [2.23]$$

$$\theta_+ = \sin^{-1}\left[\frac{r\sin\theta}{\sqrt{r^2 + d^2 - 2rd\cos\theta}}\right], \qquad [2.24]$$

$$r_-^2 = r^2 + d^2 + 2rd\cos\theta, \qquad [2.25]$$

and

$$\theta_- = \sin^{-1}\left[\frac{r\sin\theta}{\sqrt{r^2 + d^2 + 2rd\cos\theta}}\right], \qquad [2.26]$$

The magnetic field corresponding to the total vector potential is

$$B_r = \frac{1}{r\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta A_\phi\right), \qquad [2.27a]$$

$$B_\theta = -\frac{1}{r}\frac{\partial}{\partial r}\left(rA_\phi\right), \qquad [2.27b]$$

$$B_\phi = 0. \qquad [2.27c]$$

The procedure now involves manipulating the loop variables until Eqs. [2.27] describe a magnetic bottle field gradient (Eq. [2.13]).

The constraint applied enforces the magnetic bottle shape over the plane of the central loop, thus requiring that

$$B_z\left(\theta=\frac{\pi}{2}\right)=B_2\left(z^2-\frac{x^2+y^2}{2}\right)\Bigg|_{\theta=\frac{\pi}{2}}=B_2\left(-\frac{r^2}{2}\right). \qquad [2.28]$$

The calculation is carried out near the center of the loop assembly, and B_z is thus expanded in r about r = 0, for small r ($r \ll R_1$ and $r \ll R_0$), yielding

$$B_z\left(\theta=\frac{\pi}{2}, r\to 0\right)=\frac{2\pi}{c}\left[\frac{I_0}{R_0}+\frac{2I_1R_1^2}{\left(d^2+R_1^2\right)^{3/2}}\right]r^0-\frac{9\pi}{c}\left[\frac{I_0}{R_0^2}+\frac{2I_1R_1^3}{\left(d^2+R_1^2\right)^{5/2}}\right]r^1$$

$$+\frac{12\pi I_0}{cR_0^3}\left[2-\left(\frac{I_1}{I_0}\right)\sqrt{\frac{R_0}{R_1}}+5\left(\frac{I_1}{I_0}\right)\sqrt{\frac{R_1}{R_0}}\right]r^2+O\left(r^3\right). \qquad [2.29]$$

According to the constraint, B_z should have no r^0 or r^1 terms. Setting the r^0 coefficient to zero yields the condition

$$I_1=-I_0\frac{\left(R_1^2+d^2\right)^{3/2}}{2R_0R_1^2}. \qquad [2.30]$$

Setting the r^1 coefficient equal to 0, solving for I_1, and equating the result to the expression above produces the second condition

$$d^2=R_0R_1-R_1^2. \qquad [2.31]$$

Since d^2, R_0, and R_1 are all positive numbers, $R_0 > R_1$. Combining the expressions for I_1 and d^2 results in

$$I_1=-\frac{1}{2}I_0\sqrt{\frac{R_0}{R_1}}. \qquad [2.32]$$

This expression may be inserted into the r^2 coefficient; setting the coefficient equal to $-B_2/2$ and solving for B_2 gives the magnetic bottle strength

$$B_2=\frac{12\pi I_0}{cR_0^2}\left(\frac{1}{R_0}-\frac{1}{R_1}\right), \qquad [2.33]$$

whose sign is opposite to that of the current I_0.

The design is made easier to construct in practice by using the same current magnitude in all three loops (with opposite orientation in the end loops relative to that in the central loop): $I_1 = -I_0$. This sets the loop radii at $R_1 = 0.25R_0$, with the end loops separated from the central loop by $d = 0.25\sqrt{3}R_0 = 0.433R_0$. Under these conditions, the magnetic bottle strength is

$$B_2 = \frac{-36\pi I_0}{cR_0^3}. \qquad [2.34].$$

A likely design would call for $R_0 = 1$ cm, $R_1 = 0.25$ cm, and $d = 0.43$ cm; with a 10 A current, this would yield a bottle strength of 113 G/cm^2 for each turn of wire used, requiring nine turns to reach 1000 G/cm^2. Note that, for resistive wire, there is a tradeoff between cooling requirements and the validity of the approximations used. Specifically, bundles of thinner wire are more likely to resemble current loops of infinitesimal cross section, but the higher resistivity of thin wire will increase severely the amount of heat dissipated. For example, a 3×3 bundle of 20-gauge wire will have a large cross section radius of 0.12 cm, but its power consumption at 20° C will be only 27 W, whereas the small cross section radius of a 32-gauge bundle (0.03 cm) will require handling 450 W, which is feasible with a circulating coolant. (Values for wire diameters and resistivities are available in the CRC Handbook of Chemistry and Physics.[14]) Superconducting analogs of this geometry are also worth pursuing, in which case the magnetic bottle would be included in the same cryostat as the detection electronics.

2.4 Numerical simulation of classical ion trajectories

The complexity of the equations of motion [2.19] in a Penning trap and magnetic bottle does not allow a classical analytic solution of the problem. Furthermore, the trajectories of interest fall outside the range of application of the guiding center approximation,[15] especially considering the extreme accuracy needed ($\approx 10^{-7}$ in frequency).

The equations must thus be solved numerically via an iterative integration of individual ion trajectories for given sets of initial conditions. A good simulation program can be useful in determining the stability of trap configurations, extracting the frequencies of the various ion modes, and understanding the requirements and results of hypothesized experiments.

The trajectory simulations written to support much of the work in this dissertation are included in Appendix A. They allow the study of ion motions in three dimensions and can be altered to include any combination of electric and magnetic fields. Changes in motional amplitudes can then be analyzed by direct observation of Cartesian positions. In addition, Fourier transformation facilitates the extraction of motional frequencies.

2.5 Axial motion in strong magnetic bottles: numerical simulations

Simulations of ion trajectories in Penning traps with strong magnetic bottles show frequency shifts which scale with the size of the magnetic bottle strength. Table 2.1 lists the cyclotron and axial frequencies observed in trajectory simulations of a singly charged, positive, 100 amu ion in a Penning trap with B_0 = 1 T, V_0 = 10 V, and d = 1 cm. The simulations were all carried out with a frequency resolution of 15 Hz and with an integration time step of 6.9×10^{-8} s (except for the 10000 T/m^2 calculation, carried out with a 2.5 ns step). As the bottle intensity is increased, the axial transform continues to consist of a single frequency (to better than 1 %) in strong bottles ($B_2 \approx$ 1000 G/cm^2 in Penning traps with $B_0 \approx$ 1 to 5 T). This is understood qualitatively if the axial motion is assumed to remain strongly decoupled from the cyclotron and magnetron modes (a point that will be important in the experiments proposed below).

The simulation programs also reveal previously unreported behavior in the strong bottle limit: the cyclotron motion is modulated at twice the axial frequency. Small motional sidebands of the ω_+ centerband, with intensity up to $\approx$ 10 % of the centerband, appear at $\omega_+ \pm 2\omega_z$ in the Fourier transforms of the transverse coordinates (both ω_+ and

B_2 (T/m^2)	$\omega_+/2\pi$ from x FFT (Hz)	$\omega_z/2\pi$ from z FFT (Hz)	$\frac{I(\omega_+ - 2\omega_z)}{I(\omega_+)}$ in x FFT (%)	$\frac{I(\omega_+ + 2\omega_z)}{I(\omega_+)}$ in x FFT (%)	$\frac{I(\text{peak "noise"})}{I(\omega_z)}$ in z FFT (%)
100	147450	50790	1.3	0.4	0.02
1000	147465	55545	11.1	1.8	0.3
10000	146010	90270*	7.1	11.8	2.3

Table 2.1. Trapping frequencies for varying magnetic bottle field strengths. Trajectory simulation parameters are described in the text. The frequencies have a resolution of 15 Hz. Peak "noise" is the largest intensity in the axial spectrum other than the axial trapping frequency.

ω_z are shifted by the bottle). The transverse transform sidebands are easy to understand qualitatively as frequency modulation (FM) sidebands. A stable axial oscillation at frequency ω_z varies the z^2 term in the $\hat{z}$ component of the bottle field (Eq. [2.13]) at a frequency of $2\omega_z$. Since $\omega_+ \gg \omega_z$, a given cyclotron orbit averages out the transverse components of the bottle, but keeps the axial component: hence the modulation of the cyclotron frequency at twice the axial frequency. This FM model, however, grossly overestimates the sideband intensities, emphasizing the need for the quantitative trajectory simulation. The small size of the axial frequency sidebands and the absence of other modulation facilitates the development of the theory presented below. The size of the frequency shifts observed will be justified quantitatively by quantum mechanical perturbation estimates in Chapter 3.

2.6 Spin dependent axial potentials

Examination of equations of motion [2.19] in light of the decoupled axial motion observed in numerical simulations suggests a helpful simplification which will be used to derive analytical solutions for the effect of magnetic resonance on the axial motion. The only dependence of the axial equation of motion on the transverse coordinates comes in through the term with $x\frac{dy}{dt} - y\frac{dx}{dt}$, and through $|\vec{B}|$. It is interesting to note that insertion of the transverse solutions for a Penning trap with no magnetic bottle (the sum of two oscillators with frequencies ω_+ and ω_-) makes the term with derivatives vanish. In the presence of a bottle, this term must also remain small, since no modulation of the axial motion by the transverse modes is observed in simulations. This term, in fact, defines the mechanical magnetic moment μ_m of the ion,

$$\mu_m = \frac{q}{2}\left(x\frac{dy}{dt} - y\frac{dx}{dt}\right), \qquad [2.35]$$

which is an approximate constant of the motion.[15] Note that, in an ideal Penning trap with no magnetic bottle, the mechanical magnetic moment reduces to

$$\mu_m = \frac{1}{2}q\omega_+\rho_+^2 + \frac{1}{2}q\omega_-\rho_-^2. \qquad [2.36]$$

Neglecting spin, the magnetic bottle shifts the axial frequency by

$$\Delta\omega_z = \frac{\mu_m B_2}{m\omega_z}. \qquad [2.37]$$

In the limit of no magnetron motion, this expression reduces to the shift calculated by Laukien[12] and presented in Eq. [2.21]. Furthermore, in the limit where $|\Delta\vec{B}| \ll B_0$ over all points sampled by the ion trajectory (which occurs for typical orbit sizes, even for relatively large B_2), the term in the axial equation of motion which involves $|\vec{B}|$ can be expanded about $B_2 \approx 0$:

$$\frac{2\mu B_2}{m|\bar{B}|}\left(B_0 + B_2^2 z^2\right)z = \frac{2\mu B_2}{m} z - \frac{2\mu B_2^2}{mB_0} z\left(z^2 - \frac{\rho^2}{2}\right) + O\left(B_2^3\right). \qquad [2.38]$$

The term quadratic in B_2 can be neglected and helps explain the decoupled axial motion observed in the axial simulations, since the axial and transverse coordinates remain decoupled through first order in B_2. Under these conditions, the axial equation of motion becomes

$$\frac{d^2 z}{dt^2} + \left[\omega_z^2 + \frac{2B_2}{m}(\mu_m - \mu)\right]z = 0. \qquad [2.39]$$

Since the case under consideration is a spin 1/2 ion, $\mu = \pm g|\mu_B|/2$ for $m_s = \pm 1/2$. Eq. [2.39] is now split into an explicitly spin dependent equation,

$$\frac{d^2 z}{dt^2} + \left[\omega_z^2 + \frac{2B_2}{m}\left(\mu_m \mp \frac{g}{2}|\mu_B|\right)\right]z = 0 \qquad [2.40]$$

for $m_s = \pm 1/2$.

This differential equation may be solved by the energy method.[16] Straightforward integration of the equation with respect to the axial coordinate gives the potential energy per unit mass. Defining spin up and spin down potentials as U_+ and U_-, respectively, the potential per unit mass is

$$\frac{1}{m}U_\pm = \int\left\{\left[\omega_z^2 + \frac{2B_2}{m}\left(\mu_m \mp \frac{g}{2}|\mu_B|\right)\right]z\right\}dz. \qquad [2.41]$$

Performing the integration, and substituting in Eq. [2.8] for ω_z^2,

$$U_\pm = \left[\frac{qV_0}{2d^2} + \left(\mu_m \mp \frac{g}{2}|\mu_B|\right)B_2\right]z^2 \pm \frac{\hbar\omega_{L0}}{2}, \qquad [2.42]$$

where the constants of integration $\pm\frac{\hbar\omega_{L0}}{2}$ are the z=0 energies of each curve. At z=0, the only potential energy contribution arises from the Zeeman splitting of the electron spin levels in the homogeneous magnetic field $B_0\hat{z}$; thus,

$$U_+(z=0) - U_-(z=0) = \hbar\omega_{L0}. \qquad [2.43]$$

Note that these potentials are independent of mass, and $U_+ > U_-$ up to a crossing point, since U_- is steeper than U_+ over this region. The two potential curves cross at $\pm\sqrt{B_0/B_2}$; with B_0=1 T and B_2=1000 T/m^2, the crossing point occurs at 3.16 cm, much larger than typical trap dimensions, so the crossing points play no role here. The difference in axial frequencies due to the different curvatures is negligible; this will be corroborated by the negligible weight of the spin-dependent term in the quantum perturbation estimate for the axial frequency shift to be presented in Chapter 3.

The difference in curvature of the two potentials implies that a given amount of axial energy will translate into a larger axial amplitude in the shallower (spin up) curve. Following the usual notation, $|\alpha\rangle$ stands for the spin up (m_s=1/2) state, and $|\beta\rangle$ stands for the spin down (m_s=−1/2) state. Then, $|\alpha\rangle \leftrightarrow |\beta\rangle$ transitions should cause a change in the axial amplitude of a trapped ion. Consider transitions due to π pulses precisely at z=0 to maximize the effect, when all of the axial trapping energy is "stored" as kinetic energy. The ion carries this kinetic energy with it when the π pulse moves it from one potential to the other. Moving between these potentials of different curvature changes the classical turning points, i.e., the amplitude of the axial motion. Defining z_+ as the axial amplitude in state $|\alpha\rangle$, and z_- as that in state $|\beta\rangle$, the axial amplitude change for a transition is obtained by solving the equation

$$U_+(z = z_+) = U_-(z = z_-) + \hbar\omega_{L0}. \qquad [2.44]$$

This effect will be employed directly to design techniques for monitoring ESR of trapped ions in Chapter 4.

2.7 References

[1]Purcell, E.M., *Electricity and Magnetism*, 2nd ed. (McGraw-Hill, New York), 1985.

[2]Beauchamp, J.L., and J.T. Armstrong, Rev. Sci. Inst. **40**, 123 (1969).

[3]Penning, F.M., Physica (Utrecht) **3**, 873 (1936).

[4]Brown, L.S., and G.Gabrielse, Rev. Mod. Phys. **58**, 233 (1986).

[5]Van Dyck, R.S., Jr., F.L. Moore, D.L. Farnham, and P.B. Schwinberg, Rev. Sci. Inst. **57**, 593 (1986).

[6]Gerlach, W., and O. Stern, Zeits. fur Phys. **9**, 349 (1922).

[7]Cohen-Tannoudji, C., B. Diu, and F. Laloë, *Quantum Mechanics*, Vol. 1 (John Wiley & Sons, New York), 1977.

[8]Sakurai, J.J., *Modern Quantum Mechanics* (Benjamin/Cummings, Menlo Park), 1985.

[9]Van Dyck, R.S., Jr., P.B. Schwinberg, and H.G. Dehmelt, in *New Frontiers in High Energy Physics*, edited by B. Kursunoglu, A. Perlmutter, and L. Scott (Plenum, New York, 1978).

[10]Van Dyck, R.S., Jr., P.B. Schwinberg, and H.G. Dehmelt, in Atomic Physics **9**, edited by R.S. Van Dyck, Jr., and E.N. Fortson (World Scientific, Singapore, 1984).

[11]Goldstein, H., *Classical Mechanics*, 2nd ed. (Addison-Wesley, Reading), 1980.

[12]Laukien, F.H., Int. J. Mass Spectrom. Ion Processes **73**, 81 (1986).

[13]Jackson, J.D., *Classical Electrodynamics*, 2nd ed. (Wiley, New York), 1975.

[14]Weast, Robert C., ed., *CRC Handbook of Chemistry and Physics*, 63rd ed. (CRC Press, Boca Raton), 1982.

[15]Schmidt, G., *Physics of High Temperature Plasmas*, 2nd ed. (Academic Press, New York), 1979.

[16]Sanchez, D.A., R.C. Allen, and W.T. Kyner, *Differential Equations* (Addison-Wesley, Reading, MA), 1983.

Chapter 3: Quantum Mechanical Description of Ion Motion in a Penning Trap

3.1 Introduction

A quantum mechanical description of motion in the Penning trap is possible. This advantageous tool yields new insight into the motion and makes perturbation estimates of modifications due to more complex fields readily available. An early motivation for a quantum treatment was that an electron or positron at a temperature of 4 K is not a classical particle and, in particular, its cyclotron motion (more so than the magnetron and axial motions) must be described quantum mechanically.[1] Since the research proposed in this dissertation deals with molecular ions, the need for a quantum mechanical formalism must be justified. Other workers[2,3] have described ICR by building classical oscillators from superpositions of quantum mechanical states. The approach followed here relies only on energy eigenstates to yield perturbation results. The classical limit is obtained by considering large quantum numbers characteristic of the kinetic energies of interest. This provides an easy mechanism to calculate the perturbation of ion motions by modifications of the electric and magnetic fields. This is illustrated by the calculation of the radial and axial dependence of cyclotron frequency shifts due to the addition of a magnetic bottle field. These and other quantum results can be checked against numerical classical simulations, providing corroboration of each method. Thus, quantum and classical mechanics become complementary techniques in the study of trapped ion motions.

3.2 *Solutions for the axial and transverse modes*

Brown and Gabrielse[1] develop a quantum mechanical treatment that relies on the raising and lowering operators for the axial, cyclotron, and magnetron degrees of freedom. Since the motions are harmonic, the solution of the quantum harmonic oscillator is readily applied to each case. It is best to begin with the axial motion, since it is an exact harmonic oscillator in one dimension, with Hamiltonian

$$H_z = \frac{p_z^2}{2m} + \frac{m\omega_z^2 z^2}{2}, \qquad [3.1]$$

and canonical commutation relation

$$[z, p_z] = i\hbar. \qquad [3.2]$$

The usual creation and annihilation operators are

$$a_z^\dagger = \sqrt{\frac{m\omega_z}{2\hbar}}z - i\sqrt{\frac{1}{2m\hbar\omega_z}}p_z, \qquad [3.3a]$$

$$a_z = \sqrt{\frac{m\omega_z}{2\hbar}}z + i\sqrt{\frac{1}{2m\hbar\omega_z}}p_z, \qquad [3.3b]$$

with

$$\left[a_z, a_z^\dagger\right] = 1. \qquad [3.4]$$

Inverting equations [3.3],

$$z = \sqrt{\frac{\hbar}{2m\omega_z}}\left(a_z^\dagger + a_z\right), \qquad [3.5a]$$

$$p_z = i\sqrt{\frac{m\hbar\omega_z}{2}}\left(a_z^\dagger - a_z\right). \qquad [3.5b]$$

Substituting equations [3.5] into the Hamiltonian [3.1] yields

$$H_z = \hbar\omega_z\left(a_z^\dagger a_z + \frac{1}{2}\right) \qquad [3.6]$$

with the usual orthonormal energy eigenkets $|k\rangle$, k=0,1,2,... with corresponding energies

$$E_z = \hbar\omega_z\left(k+\frac{1}{2}\right) \qquad [3.7]$$

and the creation and annihilation relations

$$a_z|k\rangle = \sqrt{k}\,|k-1\rangle \qquad [3.8a]$$

$$a_z^\dagger|k\rangle = \sqrt{k+1}\,|k+1\rangle. \qquad [3.8b]$$

The transverse Hamiltonian is quantized in complete analogy to the axial Hamiltonian. Cyclotron (+) and magnetron (–) creation and destruction operators are defined from the transverse vectors of equation [2.10]:

$$a_\pm = \sqrt{\frac{m}{2\hbar(\omega_+-\omega_-)}}\left(V_x^{(\pm)} \mp iV_y^{(\pm)}\right); \qquad [3.9a]$$

$$a_\pm^\dagger = \sqrt{\frac{m}{2\hbar(\omega_+-\omega_-)}}\left(V_x^{(\pm)} \pm iV_y^{(\pm)}\right). \qquad [3.9b]$$

Inserting these into the classical Hamiltonian [2.11] reveals the quantum Hamiltonian:

$$H_\rho = \hbar\omega_+\left(a_+^\dagger a_+ + \frac{1}{2}\right) - \hbar\omega_-\left(a_-^\dagger a_- + \frac{1}{2}\right). \qquad [3.10]$$

H_ρ is separable into its cyclotron and magnetron components; thus, the transverse energy is the sum of the cyclotron and magnetron energies, and its eigenstates are the direct products of the cyclotron and magnetron energy eigenstates. Using the index n for the cyclotron motion and the index l for the magnetron motion,

$$E_{n,l} = \hbar\omega_+\left(n+\frac{1}{2}\right) - \hbar\omega_-\left(l+\frac{1}{2}\right); \qquad [3.11]$$

$$|n\,l\rangle = |n\rangle\otimes|l\rangle, \qquad [3.12]$$

with creation and annihilation relations analogous to equations [3.8]. The total Hamiltonian for an ion in a Penning trap is

$$H = H_z + H_\rho, \qquad [3.13]$$

and the total energy is

$$E = E_k + E_{n,l}. \qquad [3.14]$$

The transverse coordinates can be written in terms of the operators [3.9]. They depend on the sign of the ionic charge, sign(q), through equation [2.10], which defines the transverse vectors $\bar{V}^{(\pm)}$. Throughout the remainder of the dissertation, results will be presented for the case of a positively charged ion, with sign(q) = 1. The transverse coordinates are

$$x = i\sqrt{\frac{\hbar}{2m(\omega_+ - \omega_-)}}\left(a_+^\dagger - a_+ + a_-^\dagger - a_-\right), \qquad [3.15a]$$

$$y = \sqrt{\frac{\hbar}{2m(\omega_+ - \omega_-)}}\left(a_+^\dagger + a_+ - a_-^\dagger - a_-\right), \qquad [3.15b]$$

$$\rho = x + iy = i\sqrt{\frac{2\hbar}{m(\omega_+ - \omega_-)}}\left(a_+^\dagger - a_-\right), \qquad [3.15c]$$

$$\rho^2 = \frac{2\hbar}{m(\omega_+ - \omega_-)}\left(a_+^\dagger a_+ + a_-^\dagger a_- + 1 - a_+^\dagger a_-^\dagger - a_+ a_-\right), \qquad [3.15d]$$

and the kinetic momenta are

$$p_x = \sqrt{\frac{\hbar m}{2(\omega_+ - \omega_-)}}\left[\omega_+\left(a_+^\dagger + a_+\right) - \omega_-\left(a_-^\dagger + a_-\right)\right], \qquad [3.16a]$$

$$p_y = -i\sqrt{\frac{\hbar m}{2(\omega_+ - \omega_-)}}\left[\omega_+\left(a_+^\dagger - a_+\right) + \omega_-\left(a_-^\dagger - a_-\right)\right]. \qquad [3.16b]$$

Equations [3.5], [3.15], and [3.16] can be used to represent any physical observable in terms of the axial, cyclotron, and magnetron creation and annihilation operators. Many typical computations are facilitated by this method's commutation relations, as will become apparent in the following section: most commutators vanish, and the only ones that survive are

$$\left[a_i, a_i^\dagger\right] = 1, \quad i = +, -, k. \qquad [3.17]$$

3.3 Perturbation calculation of frequency shifts due to a magnetic bottle

The magnetic bottle was introduced in Chapter 2 (Eqn. [2.13]) as a tool to couple spin to trapped ion motions. Semiclassical estimates and the results of numerical simulations were presented to describe its effect on motional frequencies. Quantum mechanical perturbation theory is useful here, as long as $|\Delta\vec{B}| \ll B_0$ over the ion orbit. The magnetic bottle adds a perturbation Hamiltonian ΔH to the Penning trap Hamiltonian [3.13]. As before, it is convenient to restrict this discussion to the case of a spin 1/2 ion; the results can be extended easily to ions of different spin. The ion will once again be considered a point particle with mass m and spin 1/2 due to a paramagnetic electron, with magnetic moment $\pm g|\mu_B|/2$, where μ_B is the Bohr magneton. If $\vec{\sigma}$ represents the Pauli spin matrices, the perturbation Hamiltonian is[1]

$$\Delta H = \frac{g|\mu_B|}{2}\vec{\sigma}\bullet\Delta\vec{B} - q\vec{v}\bullet\Delta\vec{A}. \qquad [3.18]$$

$\Delta\vec{A}$ is the modification to the vector potential. In the symmetric gauge, where the vector potential for the homogeneous magnetic field of the Penning trap is

$$\vec{A} = \frac{1}{2}\vec{B}\times\vec{\rho}, \qquad [3.19]$$

the modification to the vector potential due to the magnetic bottle is

$$\Delta\vec{A} = \frac{1}{2}B_2\left(z^2 - \frac{\rho^2}{4}\right)\hat{B}\times\vec{\rho}. \qquad [3.20]$$

Since spin is now included in the problem, spin wavefunctions must be accounted for. The eigenstates $|nkls\rangle$ of the total system in the absence of the magnetic bottle are direct product states of the cyclotron, axial, magnetron, and spin eigenstates:

$$|nkls\rangle = |n\rangle\otimes|k\rangle\otimes|l\rangle\otimes|s\rangle. \qquad [3.21]$$

Likewise, the total energy $E_{n,k,l,s}$ is the sum

$$E_{n,k,l,s} = \hbar\omega_+\left(n+\tfrac{1}{2}\right) - \hbar\omega_-\left(l+\tfrac{1}{2}\right) + \hbar\omega_z\left(k+\tfrac{1}{2}\right) + \hbar\omega_{L0}m_s, \qquad [3.22]$$

with $m_s = \pm 1/2$, and ω_{L0} the Larmor frequency of the electron,

$$\omega_{L0} = \frac{g|\mu_B|B_0}{\hbar}. \qquad [3.23]$$

Perturbation theory now gives the first order correction to the energy as[4]

$$\Delta E_{n,k,l,s} = \langle nkls|\Delta H|nkls\rangle. \qquad [3.24]$$

First order frequency shifts associated with quantum number i are then calculated by holding the other three quantum numbers constant and subtracting two adjacent perturbed energy levels of i:

$$\Delta\omega_i = \frac{1}{\hbar}(\Delta E_{i+1} - \Delta E_i). \qquad [3.25]$$

The shifts in the cyclotron, magnetron, axial, and Larmor frequencies are

$$\Delta\omega_+ = \frac{B_2}{m(\omega_+ - \omega_-)} \times$$

$$\left\{|\mu_B|gm_s + \frac{q\hbar}{m}\left[\frac{\omega_+}{\omega_z}\left(k+\tfrac{1}{2}\right) - \frac{\omega_+ + \omega_-}{\omega_+ - \omega_-}\left(l+\tfrac{1}{2}\right) - \frac{\omega_+}{\omega_+ - \omega_-}(n+1)\right]\right\}, \qquad [3.26a]$$

$$\Delta\omega_- = \frac{B_2}{m(\omega_+ - \omega_-)} \times$$

$$\left\{|\mu_B|gm_s + \frac{q\hbar}{m}\left[\frac{\omega_-}{\omega_z}\left(k+\tfrac{1}{2}\right) - \frac{\omega_+ + \omega_-}{\omega_+ - \omega_-}\left(n+\tfrac{1}{2}\right) - \frac{\omega_-}{\omega_+ - \omega_-}(l+1)\right]\right\} \qquad [3.26b]$$

$$\Delta\omega_z = \frac{B_2}{m\omega_z}\left\{-|\mu_B|gm_s + \frac{q\hbar}{m}\left[\frac{\omega_+}{\omega_+ - \omega_-}\left(n+\tfrac{1}{2}\right) + \frac{\omega_-}{\omega_+ - \omega_-}\left(l+\tfrac{1}{2}\right)\right]\right\}, \qquad [3.26c]$$

$$\Delta\omega_{L0} = -\frac{B_2 g|\mu_B|}{2m}\left[\frac{1}{\omega_z}\left(k+\tfrac{1}{2}\right) - \frac{1}{\omega_+ - \omega_-}(n+l+1)\right]. \qquad [3.26d]$$

The magnetic bottle couples each degree of freedom to the other three. Changes in a given quantum number may be monitored by changes in the observed frequency

associated with another. This is useful, for example, in the single electron experiments, which made use of the first term in $\Delta\omega_z$.[1] In a typical laboratory field of 5 T, the electron cyclotron frequency ν_+ is about 160 GHz, a microwave frequency too high to be observed easily with the current technology. The axial frequency ν_z (with $V_0 \approx 10$ V and $d \approx .3$ cm) is approximately 60 MHz, an easily observable radiofrequency. A weak magnetic bottle is thus added, coupling the cyclotron motion to the axial oscillation. Changes in the cyclotron motion, as well as spin flips, are thus monitored by measuring changes in the axial frequency.[1]

A quantum mechanical analogue of the classical cyclotron frequency shift [2.20] may now be derived. The quantum analogue of the classical radius (squared) of state $|nkls\rangle$ is the expectation value of the operator for ρ^2 (Eq. [3.15c]),

$$\rho^2 = \langle nkls|\rho^2|nkls\rangle = \frac{2\hbar}{m(\omega_+ - \omega_-)}(n + l + 1). \qquad [3.27]$$

Since $\omega_+ \gg \omega_z \gg \omega_-$, and in the limit of small k, the cyclotron frequency shift [3.26a] is approximately

$$\Delta\omega_+ \approx -\frac{q}{2m} B_2 \rho^2. \qquad [3.28]$$

This quantum perturbation expression is equal to the classical shift [2.20]. However, the quantum expression is also valid for arbitrary k, a fact that will be important for the analysis of experiments proposed later in the dissertation. This functional dependence will be exploited to detect small changes in the axial quantum number as cyclotron frequency shifts in the presence of a magnetic bottle.

3.4 Rotating frame calculation of ICR excitation

The previous section employed perturbation theory to study the effects of a static magnetic field gradient. The quantum mechanical formalism is also useful for calculating

the behavior of trapped ions subject to oscillating magnetic and electric fields. Such calculations are performed in a rotating frame defined by the unperturbed ion modes. To illustrate the method, the cyclotron radius change due to ICR excitation by a resonant electric field is calculated in this section.

The calculation begins by using the transverse part H_ρ of the Penning trap Hamiltonian (Eq. [3.10]) to define a rotating frame. The lab frame perturbation due to the ICR electric field is

$$V = qEx\cos(\omega t). \qquad [3.29]$$

This can be rewritten in terms of quantum operators by using the (q > 0) operator expression for x, Eq. [3.15a], to yield

$$V = iqE\sqrt{\frac{\hbar}{2m(\omega_+ - \omega_-)}}\left(a_+^\dagger - a_+ + a_-^\dagger - a_-\right)\cos(\omega t). \qquad [3.30]$$

The perturbation is transformed to the rotating frame by expanding the expression

$$\tilde{V} = e^{-iH_\rho t/\hbar} V e^{iH_\rho t/\hbar}. \qquad [3.31]$$

via the Baker-Haussdorf formula[4]

$$e^{iG\lambda} A e^{-iG\lambda} = A + i\lambda[G,A],$$

$$+\left(\frac{i^2\lambda^2}{2!}\right)[G,[G,A]]+\cdots+\left(\frac{i^n\lambda^n}{n!}\right)[G,[G,\ldots[G,A]]\ldots] \qquad [3.32]$$

where G is a Hermitian operator and λ is a real parameter. The resulting rotating frame Hamiltonian is

$$V = \frac{i}{2}qE\sqrt{\frac{\hbar}{2m(\omega_+ - \omega_-)}} \times$$

$$\left\{a_+^\dagger e^{i(\omega-\omega_+)t} - a_+ e^{-i(\omega-\omega_+)t} + a_+^\dagger e^{-i(\omega+\omega_+)t} - a_+ e^{i(\omega+\omega_+)t}\right.$$

$$+a_-^\dagger e^{-i(\omega-\omega_-)t} - a_- e^{i(\omega-\omega_-)t} + a_-^\dagger e^{i(\omega+\omega_-)t} - a_- e^{-i(\omega+\omega_-)t}\Big\}. \quad [3.33]$$

Setting $\omega = \omega_+$ to meet the ion cyclotron resonance condition and keeping only those terms which are constant in the rotating frame, the average rotating frame Hamiltonian becomes

$$\tilde{V}^{(0)} = \frac{i}{2} qE \sqrt{\frac{\hbar}{2m(\omega_+ - \omega_-)}} \left(a_+^\dagger - a_+\right). \quad [3.34]$$

The goal of the calculation is to determine how the radius operator ρ changes with time upon application of the resonant electric field; this is solved via

$$\rho_{ICR}(t) = \langle \Psi | e^{-i\tilde{V}^{(0)}t/\hbar} e^{-iH_\rho t/\hbar} \rho e^{i\tilde{V}^{(0)}t/\hbar} e^{iH_\rho t/\hbar} | \Psi \rangle. \quad [3.35]$$

The initial wavefunction $|\Psi\rangle$ is the quantum mechanical representation of the (classical) trapped ion motion, and may be solved for by calculating $\rho(t)$, the position in the absence of any perturbation, to yield normal cyclotron and magnetron motion. This derivation starts with the general wavefunction

$$|\Psi\rangle = \sum_{n,l} |c_n| e^{i\phi_n} |c_l| e^{i\phi_l} |n,l\rangle, \quad [3.36]$$

where the coefficient associated with the cyclotron ket $|n\rangle$ is written explicitly in terms of its magnitude $|c_n|$ and its phase ϕ_n, and similarly for the magnetron ket. The unperturbed position is

$$\rho(t) = \langle \Psi | e^{-iH_\rho t/\hbar} \rho e^{iH_\rho t/\hbar} | \Psi \rangle. \quad [3.37]$$

Substituting in the expression for the wavefunction and transforming ρ into the rotating frame,

$$\rho(t) = \left(\sum_{n',l'} |c_{n'}| e^{-i\phi_{n'}} |c_{l'}| e^{-i\phi_{l'}} \langle n',l'| \right) \left[i \sqrt{\frac{2\hbar}{m(\omega_+ - \omega_-)}} \left(a_+^\dagger e^{-i\omega_+ t} - a_- e^{-i\omega_- t}\right) \right]$$

$$\times \left(\sum_{n,l} |c_n| e^{i\phi_n} |c_l| e^{i\phi_l} |n,l\rangle \right). \quad [3.38]$$

This expression reduces to

$$\rho(t) = i\sqrt{\frac{2\hbar}{m(\omega_+ - \omega_-)}}\left[\sum_n\left(|c_{n+1}||c_n|e^{i(\phi_n - \phi_{n+1})}\sqrt{n+1}\right)e^{-i\omega_+ t} - \sum_n\left(|c_{l-1}||c_l|e^{i(\phi_l - \phi_{l-1})}\sqrt{l}\right)e^{-i\omega_- t}\right]. \quad [3.39]$$

The final solution for the wavefunction now rests on a choice of phase and a classical approximation. By definition, both the cyclotron and magnetron phases are chosen to be zero along the positive **x** axis, requiring that the overall motion have the form

$$\rho(t) = \rho_+ e^{-i\omega_+ t} + \rho_- e^{-i\omega_- t}, \quad [3.40]$$

where ρ_+ and ρ_- are the cyclotron and magnetron radii, respectively; note that the sense of rotation is chosen to be clockwise as observed looking along the **–z** axis, as expected for a positively charged ion. Enforcing this condition sets the cyclotron and magnetron ket phases:

$$\phi_{n+1} - \phi_n = \frac{\pi}{2}; \quad [3.41a]$$

$$\phi_l - \phi_{l-1} = \frac{\pi}{2}. \quad [3.41b]$$

Since the wavefunction is to describe classical motion, it is reasonable to assume that the probability amplitude distributions for each quantum number will be nonzero over broad ranges, peaking at large values, and will therefore be smooth, allowing the assumptions

$$|c_{n+1}| \approx |c_n|; \quad [3.42a]$$

$$|c_{l+1}| \approx |c_l|; \quad [3.42b]$$

$$n \approx n+1; \quad [3.42c]$$

$$l \approx l+1. \quad [3.42d]$$

Equations [3.41] and [3.42] fully determine the wavefunction $|\Psi\rangle$. In addition, using these assumptions to compare Eqs. [3.39] and [3.40] for $\rho(t)$ enables the determination of the cyclotron and magnetron radii in terms of the average quantum numbers $\bar{n}$ and $\bar{l}$:

$$\rho_+ = \sqrt{\frac{2\hbar}{m(\omega_+ - \omega_-)}}\sqrt{\bar{n}}; \qquad [3.43a]$$

$$\rho_- = \sqrt{\frac{2\hbar}{m(\omega_+ - \omega_-)}}\sqrt{\bar{l}}. \qquad [3.43b]$$

With the correct wavefunction in hand, Eq. [3.35] for $\rho_{ICR}(t)$ can be solved by first propagating the rotating frame radius under $\tilde{V}^{(0)}$ (Heisenberg picture):

$$\rho_{ICR}(t) = \langle\Psi|\left[i\sqrt{\frac{2\hbar}{m(\omega_+ - \omega_-)}}\left(a_+^\dagger e^{-i\omega_+ t} - a_- e^{-i\omega_- t}\right)\right]|\Psi\rangle$$

$$-\frac{i}{\hbar}t\langle\Psi|\left[\frac{qE\hbar}{2m(\omega_+ - \omega_-)}e^{-i\omega_+ t}\right]|\Psi\rangle. \qquad [3.44]$$

Inserting the wavefunction and solving the problem,

$$\rho_{ICR}(t) = \left(\rho_+ - \frac{qEt}{2m(\omega_+ - \omega_-)}\right)e^{-i\omega_+ t} + \rho_- e^{-i\omega_- t}. \qquad [3.45]$$

The cyclotron radius increases linearly with time for $qE < 0$, but the magnetron radius is not affected by the resonant interaction. Using the approximation $\omega_+ - \omega_- \approx \omega_c = qB_0/m$, the radius change can be written

$$\Delta\rho_{ICR}(t) = \left|-\frac{qEt}{2m(\omega_+ - \omega_-)}\right| \approx \frac{|E|t}{2B_0}, \qquad [3.46]$$

the well-known result from the ICR literature.[5]

3.5 References

[1]Brown, L.S., and G.Gabrielse, Rev. Mod. Phys. **58**, 233 (1986).

[2]Hartmann, H., and K.-M. Chung, Theoret. Chim. Acta (Berl.) **45**, 137 (1977).

[3]Schueh, D., K.-M. Chung, and H. Hartmann, Lect. Notes in Chem. **31**, 514 (1982).

[4]Sakurai, J.J., *Modern Quantum Mechanics* (Benjamin/Cummings, Menlo Park), 1985.

[5]Comisarow, M.B., Lect. Notes in Chem. **31**, 484 (1982).

Chapter 4: Proposals for ESR Spectroscopy of Trapped Ions

4.1 Direct monitoring of ESR in an ion trap

The first experiment proposed is a general method to obtain the ESR spectrum of any small ion with unprecedented resolution under collisionless conditions in a Penning or similar ion trap. By encoding ESR transitions into axial frequency shifts of dilute trapped ions, spectra will be obtained with a sensitivity increase of at least ten orders of magnitude relative to conventional ESR and with unprecedented resolution.[1] The experiment exploits the difference in curvature (and thus axial frequency) between the spin up and spin down axial potentials presented in Chapter 2. As has been shown, this arises because the presence of the magnetic bottle $\Delta\vec{B}$ couples together the axial, cyclotron and magnetron motions and the spin.

A key illustration of this coupling was the use of the spin-dependent shift of the axial frequency to measure the g-factor of the electron[2,3,4,5,6] cooled at 4 K to the ground state of its cyclotron motion. This seminal experiment of Dehmelt observed random spin flips of a single trapped electron as spin-dependent 1 Hz shifts of the 60 MHz axial frequency, as predicted by quantum perturbation theory (Eq. [3.26c]). The straightforward generalization of this experiment to atomic and molecular ions is not practical, however, since the spin-dependent shift,

$$\Delta\omega_z(m_s) = -\frac{|\mu_B|gB_2 m_s}{m\omega_z} = -\frac{|\mu_B|gB_2 m_s d}{\sqrt{qV_0 m}}, \qquad [4.1]$$

is inversely proportional to the square root of mass. Using similar trapping values for 100 amu ions, the axial frequency shift due to an electron spin magnetic moment would be 2 mHz, and that due to a proton magnetic moment would measure only 4 μHz. Even with significant technical improvements, including the increase of the magnetic bottle strength from 150 T/m^2 to 5000 T/m^2 and the lowering of the axial frequency to 20 kHz, the ESR shift would be 0.6 Hz, while the NMR shift would remain an impractically small 0.9 mHz.

A more difficult problem is that the three trapping frequencies and the Larmor frequency become inhomogeneously broadened due to the many thermally accessible states. The *n*, *l*, and *k* contributions to the perturbed trapping frequencies of Eq. [3.26] are significantly larger than the m_s contribution, obscuring any hope of direct detection of spin-dependent behavior over an ion ensemble. Minimizing this by ion cooling methods would be time-consuming and reaching a single state of the cyclotron motion is generally impractical, since this quantum decreases inversely with mass.

The approach proposed in this section is to overcome the inhomogeneous broadening of the axial motion by convolving spectra taken before and after the spin flip. The convolution process will be shown to isolate the spin-dependent behavior for each ion within a thermally broadened ensemble. Inhomogeneity of the ESR spectrum is avoided by switching off the magnetic bottle during the spin evolution in the proposed experimental sequence.

4.1.a Proposed experimental sequence

Fig. 4.1 is the time line for the proposed experiment. Ions are created in the cell by an ionizing beam along the **z** axis or, alternatively, are guided into the cell in such a way that the trapped orbits have small cyclotron and magnetron radii.[7] The magnetic bottle field is on by this point, with $B_2<0$, and the potential difference $S^b(\tau_b)$ (b for "before") between the end electrodes due to the axial motion of the trapped ions is recorded for a period τ_{max} comparable to or longer than the inverse of the spin-dependent part of the

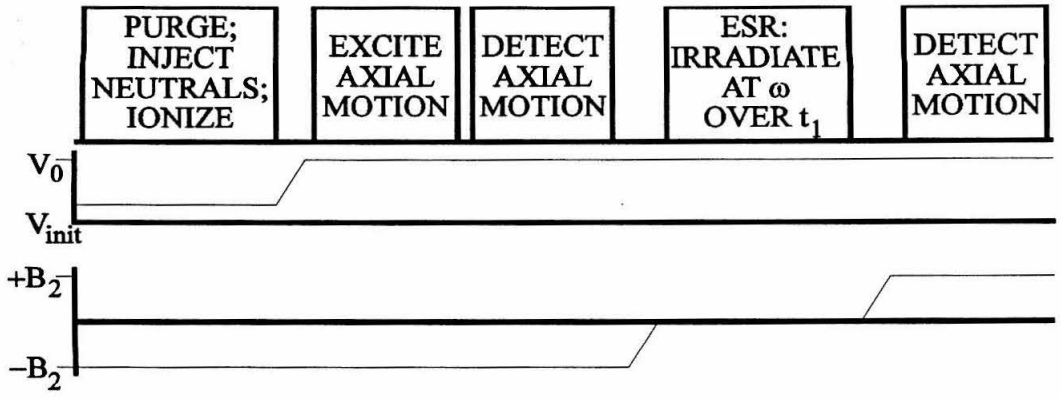


FIG. 4.1. Proposed experimental sequence, as discussed in the text. Times are not to scale.

axial frequency. The bottle field is then turned off adiabatically, i.e., slowly enough to conserve the axial action and the mechanical and spin magnetic moments. The ESR may now be encoded under homogeneous field conditions. Ideally, this is done in the time domain, to allow a high average probability of a spin flip over a large range of ESR frequencies without power broadening. Alternatively, as in the example depicted, a frequency-domain continuous wave spectrum is produced by weak microwave irradiation during a fixed time t_1. The microwave frequency ω is incremented on successive repetitions ("shots") to encode the ESR spectrum. Subsequent to the ESR period, the magnetic bottle field is restored adiabatically, but with the opposite sign. Again the axial voltage is recorded for a period τ_{max} and stored as $S^a(\tau_a)$ (a for "after").

4.1.b Signal processing algorithm: time-reversed product

The resulting data set consists of two 2D sets $S^{\zeta}(\tau_{\zeta},\omega)$, where ζ indicates b or a. Each stored axial signal is the sum of signals from individual ions at frequencies $\omega_{zn}^{\zeta\pm}$. This section will detail a procedure for processing this data set to extract the spin-dependent axial frequency differences between the "before" and "after" ESR signals; the

procedure is diagrammed in Fig. 4.2. Denote the signal from the n^{th} ion (in axial amplitude units) as

$$S_n^{\zeta}(\tau_\zeta,\omega)=\left|z_n^{\zeta}\right|\sin\left(\omega_{zn}^{\zeta\pm}\tau_\zeta+\phi_n^{\zeta}\right)\exp\left(-\gamma_z\tau_\zeta\right) \qquad [4.2]$$

where $\left|z_n^{\zeta}\right|$ is the initial magnitude of the axial turning points and the axial phase is

$$\phi_n^{\zeta}=\tan^{-1}\left[-v_{zn}(\tau_\zeta=0)/\omega_{zn}^{\zeta\pm}z_n(\tau_\zeta=0)\right]. \qquad [4.3]$$

The upper and lower signs again represent the eigenvalues of spin. The decay of the axial oscillation due to coupling to a detection circuit is represented by the damping constant γ_z.[2] The axial frequency for each ion, including corrections to ω_z due to a negative magnetic bottle, can be written from Eq. [2.42] as

$$\omega_{zn}^{b\pm}=\left\{\frac{1}{m}\left[\frac{qV_0}{d^2}-2\left(\mu_m\mp\frac{g}{2}|\mu_B|\right)|B_2|\right]\right\}^{1/2} \qquad [4.4a]$$

$$=\omega_z-\frac{|B_2|}{m\omega_z}\left(\mu_m\mp\frac{g}{2}|\mu_B|\right)+O\left(B_2^2\right). \qquad [4.4b]$$

The corresponding quantity $\omega_{zn}^{a\pm}$ for the second detection period is obtained by reversing the sign of B_2:

$$\omega_{zn}^{a\pm}=\omega_z+\frac{|B_2|}{m\omega_z}\left(\mu_m\mp\frac{g}{2}|\mu_B|\right)+O\left(B_2^2\right). \qquad [4.5]$$

The calculation will focus on values small enough that the terms through first order in B_2 suffice. Therefore, in the absence of a spin flip, the axial phase due to B_2 accumulated during τ_b is removed during τ_a. By the same reasoning, the point $\tau_a = 0$ can be chosen to begin at an interval Δt subsequent to the end of recording in τ_b so that phase accumulation due to the rise and fall transients in B_2 in this interval is zero. This leaves a net phase $\theta_0 = \omega_z \Delta t$ at $\tau_a = 0$ independent of ion orbit, which would be present even in the absence of B_2. Thus, in the absence of a spin flip, and including the decay of the orbit during the first detection period,

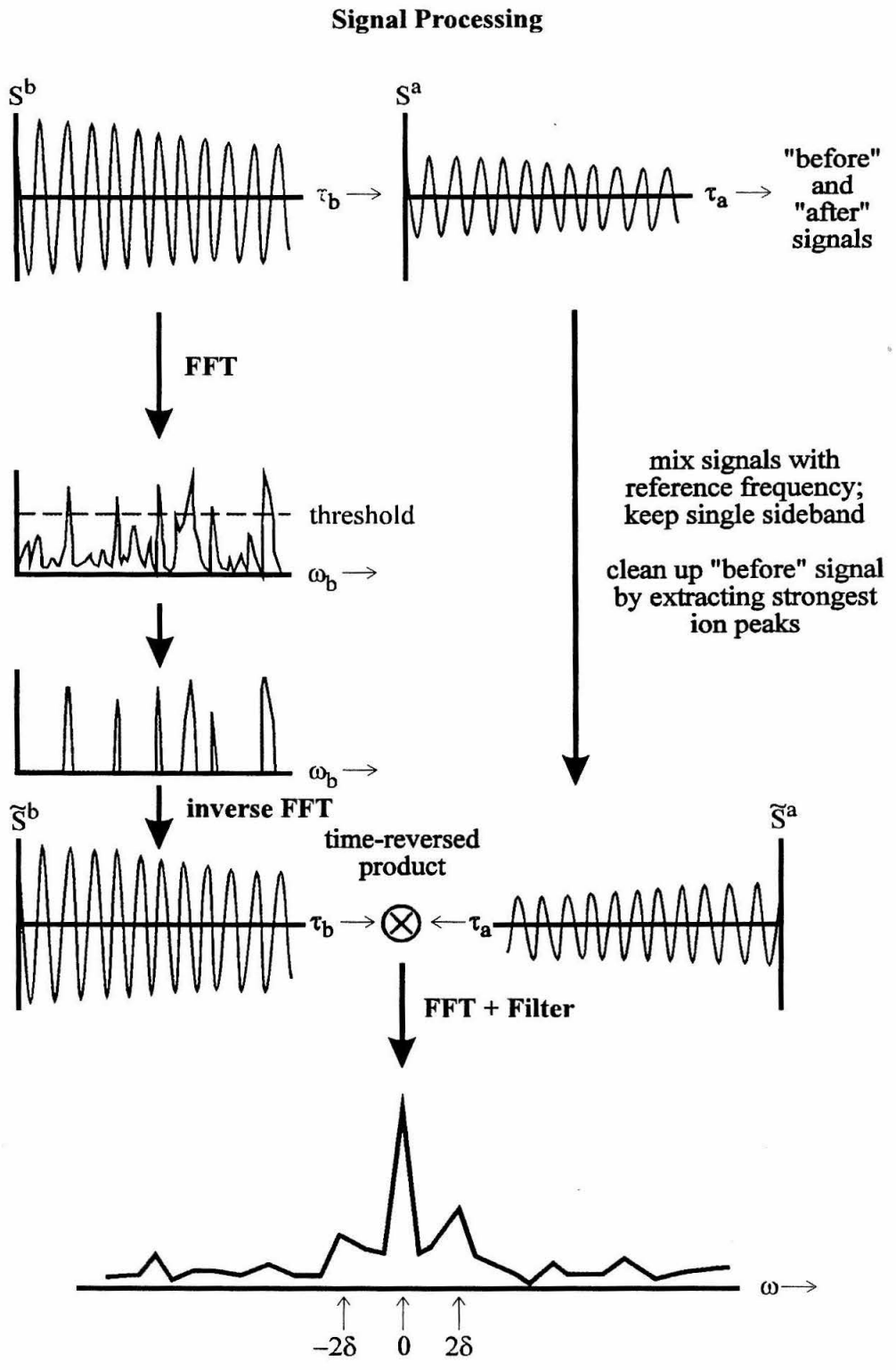


Fig. 4.2. Signal processing procedure, as described in the text.

$$\left|z_n^a\right| = \left|z_n^b\right| \exp(-\gamma_z \tau_{max}) \qquad [4.6]$$

and

$$\phi_n^a = \phi_n^b + \theta_0. \qquad [4.7]$$

If the spin is flipped during the ESR period, then $\omega_{zn}^{a\pm}$ is replaced by $\omega_{zn}^{a\mp}$ and, since the spin-mediated effect of the transients of B_2 now adds constructively, an extra phase $\delta^{\pm}$ is accumulated during the period Δt between observations. This is given by

$$\delta^{\pm} = \mp \int_0^{\Delta t} \frac{g|\mu_B \| B_2(t)|}{2m\omega_z} dt, \qquad [4.8]$$

independent of ion orbit, where the upper (lower) sign indicates m_S=+1/2 (m_S=−1/2) before MR. Thus, for the case where the spin flips,

$$\phi_n^a = \phi_n^b + \theta_0 + \delta^{\pm}. \qquad [4.9]$$

In order to limit the necessary detection bandwidth, the axial signals are mixed with the reference $\exp(-i\omega_r t)$ and single-sideband-filtered to keep only the difference term. Following single-sideband mixing, the signals are

$$\tilde{S}_n^{\zeta}(\tau_\zeta, \omega) = -\frac{i}{2}\left|z_n^{\zeta}\right| \exp\left\{i\left[\left(\omega_{zn}^{\zeta\pm} - \omega_r\right)\tau_\zeta + \phi_n^{\zeta}\right]\right\} \exp\left(-\gamma_z \tau_\zeta\right). \qquad [4.10]$$

Now consider the product formed from the "before" and "after" signals of a given ion with reversal of the time ordering in the complex conjugate of the "after" signal:

$$S_n^{ba}(\tau, \omega) \equiv \tilde{S}_n^b(\tau, \omega)\left[\tilde{S}_n^a(\tau_{max} - \tau, \omega)\right]^*. \qquad [4.11]$$

There are four distinguishable paths in spin space starting in either state and ending in either. Let $p_{\pm}$ be the initial probabilities of the two spin states and $P(\omega)$ be the probability for a stimulated spin transition. With cw irradiation at frequency ω during the time t_1, a single ESR line offset from resonance by $\Delta\omega = \omega - \omega_0$ is governed by the Rabi transition probability[8]

$$P(\omega) = \frac{\omega_1^2}{\omega_1^2 + (\Delta\omega)^2} \sin^2\left(\frac{1}{2}\sqrt{\omega_1^2 + (\Delta\omega)^2}\, t_1\right); \qquad [4.12]$$

here,

$$\omega_1 = -\gamma B_1, \qquad [4.13]$$

where γ is the gyromagnetic ratio and B_1 is the circulating magnetic field strength. Then the expectation value over an ensemble of signals can be written

$$\left\langle S_n^{ba}(\tau,\omega)\right\rangle \equiv p_+\left[1 - P(\omega)\right]\left\langle S_n^{++}(\tau)\right\rangle + p_+ P(\omega)\left\langle S_n^{+-}(\tau)\right\rangle$$
$$+ p_-\left[1 - P(\omega)\right]\left\langle S_n^{--}(\tau)\right\rangle + p_- P(\omega)\left\langle S_n^{-+}(\tau)\right\rangle. \qquad [4.14]$$

Here, for example, $S_n^{+-}(\tau)$ is the product due to an ion with spin up in the "before" signal and spin down in the "after" signal. The cases where the spin does not flip during ESR give

$$S_n^{\pm\pm}(\tau) = -\frac{i}{2}\left|z_n^b\right| \exp\left\{i\left[\left(\omega_{zn}^{b\pm} - \omega_r\right)\tau + \phi_n^b\right]\right\} \exp(-\gamma_z \tau)$$
$$\times\left\{-\frac{i}{2}\left|z_n^a\right| \exp\left\{i\left[\left(\omega_{zn}^{a\pm} - \omega_r\right)(\tau_{max} - \tau) + \phi_n^a\right]\right\} \exp\left[-\gamma_z(\tau_{max} - \tau)\right]\right\}^* \qquad [4.15a]$$
$$= \frac{1}{4}\left|z_n^b\right|^2 \exp(-2\gamma_z \tau_{max}) \exp\left\{-i\left[\left(\omega_{zn}^{a\pm} - \omega_r\right)\tau_{max} + \theta_0\right]\right\}$$
$$\times \exp\left\{i\left[\left(\omega_{zn}^{b\pm} + \omega_{zn}^{a\pm} - 2\omega_r\right)\tau\right]\right\}. \qquad [4.15b]$$

The Fourier transform of this expression for the time-reversed product is similar to the convolution of the frequency domain signals that would be obtained by Fourier transformation of the "before" and "after" signals separately. The oscillatory term in τ has the valuable property that its frequency

$$\omega_{zn}^{b\pm} + \omega_{zn}^{a\pm} - 2\omega_r \cong 2(\omega_z - \omega_r) \qquad [4.16]$$

is independent of B_2 and of ion and thus of the effects of random orbits. This allows fully constructive averaging of the magnitude spectrum over shots; the randomness of the phase

(due to the range of values of $\omega_{zn}^{a\pm}$), however, limits one to a small number of contributing ions within each shot. In the case of multiple ions in the cell, cross terms between ions exist, but vanish on average and have a broad B_2-dependent spectral range. Since this overlaps the observation region $2(\omega_z-\omega_r)$, it contributes to the noise, but will be negligible relative to the thermal noise for the cases considered of small numbers of ions at a time. The fluctuation in the number of ions in the cell from shot to shot can be eliminated as a noise source by dividing on each shot by the integral of the central lineshape; frequency-domain baseline correction before this minimizes other sources of noise.

The same considerations apply to the other two paths. The important difference is that the paths which involve spin flips lead to lines with magnitudes proportional to $p_{\pm}P(\omega)$ at the orbit-independent satellite frequencies

$$\omega_{zn}^{b\pm}+\omega_{zn}^{a\mp}-2\omega_r \cong 2(\omega_z-\omega_r)\pm\frac{g|\mu_B\|B_2|}{2m\omega_z}. \qquad [4.17]$$

Defining the ESR interferogram $S(\omega)$ as the amplitude per ion of the central line subtracted from the sum of the satellites gives

$$S(\omega) = 2P(\omega)-1. \qquad [4.18]$$

Note that this is independent of $p_{\pm}$ and so has the character of a "flop-in"[9] experiment where no initial polarization is needed and spin flips in either direction contribute identically.

Noise may be suppressed in the time-reversed product processing by introducing "noise blanking." In this procedure, all "before" signals are processed by first being Fourier transformed; once in the frequency domain, the original spectrum is then "blanked" by applying a threshold function, equal to some number above the mean intensity of the spectrum. All peaks above this line are preserved, while the points with intensities below the threshold are set to zero. The blanked spectrum is then transformed back to the time domain, and subsequently multiplied by the time-reversed "after" signal,

as detailed above. The simulations presented in the next section will illustrate this procedure.

4.1.c Simulations

The proposed experiment was simulated numerically using exact 3D trajectory calculations to confirm the validity of Eq. [4.4a], and then using Eq. [4.4a] to calculate the axial signals of ions undergoing the sequence of Fig. 1. Switching times for the magnetic bottle were also examined with 3D trajectory simulations to ensure adiabaticity. These simulations helped set the switching time parameter to values sufficiently slow to minimize the perturbation of the cyclotron and axial motions. The full experimental sequence was then simulated by a program employing analytical approximations for all relevant processes.

Magnetic bottle strength (T/m^2)	Axial frequency from trajectory simulations (Hz)	Axial frequency from Eq. [4.4a] (Hz)	Relative difference (%)
10	50790	49850	-1.85
500	52980	52130	-1.60
1000	55545	54850	-1.25
1500	57923	57430	-0.85
1000	90272	90770	0.55

Table 4.1. Comparison of axial frequency observed in trajectory simulations to that calculated with Eq. [4.4a]. Simulation parameters are discussed in the text.

4.1.c (i) Trajectory simulations of axial frequency behavior

The adequacy of Eq. [4.4a] in describing the axial frequency was tested by comparing its predictions with trajectory simulations for various values of the magnetic bottle strength. Simulated axial signals were Fourier transformed to extract the axial frequency. Singly charged positive ions of 100 amu mass were simulated with 15 Hz resolution in a Penning trap with B_0 = 1 T, V_0 = 10 V, and d = 1 cm. The cyclotron radius and the axial amplitude were both 0.5 cm, with negligible energy in the magnetron mode. The results, listed in Table 4.1, show excellent agreement between trajectory simulations and the axial frequency predicted from the mechanical moment in Eq. [4.4a].

4.1.c (ii) Trajectory simulations of magnetic bottle turn-on and turn-off

A modified version of TRAJ.FOR was written to include the effects of the magnetic bottle turn-on and turn-off periods. The bottle was assumed to turn on with a rising exponential shape, while the turn-off was modeled with a decaying exponential. Ion cyclotron radii (with no magnetron motion) and axial amplitudes were compared before and after a bottle was turned on to 1000 T/m^2 and back off again over a total time Δt, where each switch (on in $\Delta t/2$, off in $\Delta t/2$) took place over ten exponential time constants $\tau_0 = \Delta t/20$. The magnetic bottle was switched on with the form

$$\Delta\vec{B}_{on} = B_2\left[\left(z^2 - \frac{\rho^2}{2}\right)\hat{z} - z\vec{\rho}\right]\left[1 - e^{-t/(\Delta t/20)}\right] \qquad [4.19]$$

with the associated electric field

$$\Delta\vec{E}_{on} = \frac{B_2}{2\tau_0}\left(z^2 - \frac{\rho^2}{4}\right)(y\hat{x} - x\hat{y})\left(1 - e^{-t/\tau_0}\right), \qquad [4.20]$$

and switched off with the form

$$\Delta\vec{B}_{off} = B_2\left[\left(z^2 - \frac{\rho^2}{2}\right)\hat{z} - z\vec{\rho}\right]e^{-(t-t/2)/(\Delta t/20)} \qquad [4.21]$$

Total time (Δt) for bottle turn-on and turn-off (μs)	Relative cyclotron radius change compared to $B_2 = 0$ run (%)	Relative axial amplitude change compared to $B_2 = 0$ run (%)
10	−0.079	32.40
100	−0.011	5.35
1000	−0.00023	0.11

Table 4.2. Cyclotron radius and axial amplitude shifts observed in simulations of magnetic bottle switching. Parameters are explained in the text.

with the associated electric field

$$\Delta\vec{E}_{off} = -\frac{B_2}{2\tau_0}\left(z^2 - \frac{\rho^2}{4}\right)(y\hat{x} - x\hat{y})\left(1 - e^{-(t-t/2)/\tau_0}\right). \qquad [4.22]$$

Since any effects would be expected to be more pronounced for lighter ions, the simulated mass was 2 amu, with $B_0 = 1$ T, $V_0 = 10$ V, and $d = 1$ cm, with initial cyclotron radii and axial amplitudes of 0.5 cm each. Results from simulations without the associated electric fields are presented in Table 4.2. The values are presented relative to similar simulation runs with $B_2 = 0$ to eliminate numerical errors. The results for $\Delta t = 10$ μs were compared to the output from a simulation which included the electric fields, and the difference between the two calculations was negligible: with the electric fields, the cyclotron radius changed by −0.000027 % (rather than −0.079 %) and the axial amplitude grew by 32.75 % (rather than 32.40 %). Since the electric field strength is inversely proportional to the switching time, the electric fields should not change the results shown in Table 4.2 appreciably for longer switching times, and were not included in the 100 μs and 1 ms runs.

The proposed experiment seeks to measure spin-dependent axial frequency shifts on the order of 1 Hz. Other experiments proposed later in this dissertation will create similarly sized, spin-dependent shifts of the cyclotron frequency. Any other systematic shifts must thus be smaller than this quantity. The radius and amplitude changes calculated here will lead to frequency shifts in the presence of a magnetic bottle, as discussed in Chapters 2 and 3. The choice of a 2 amu mass for these simulations again provides a "worst case" scenario, since heavier masses are perturbed less readily. With a 1 ms switching time, in a 1000 T/m^2 magnetic bottle, the resulting axial amplitude change corresponds to a cyclotron frequency change of 0.1 Hz (no axial frequency shift occurs to first order in B_2). The cyclotron radius change perturbs the axial frequency by 11 μHz and the cyclotron frequency by 50 mHz. These shifts are all negligible when compared to the 1 Hz shifts to be observed.

4.1.c (iii) Simulation of proposed experimental sequence

The proposed experiment was simulated with the program AXIALCWO.FOR, a predecessor of the AXIALIR8.FOR program presented in Appendix B. A positively charged ion of 100 amu mass was trapped with B_0 = 1 T, $|B_2|$ = 5000 T/m^2, V_0 = 0.5 V, d = 0.46548 cm (with axial trap length z_0 = 0.6 cm), g = 2.0023, $\mu_B = 9.285 \times 10^{-24}$ J/T, τ_{max} = 10 s, and magnetic bottle switching time Δt = 0.1 s. The axial amplitude, cyclotron and magnetron radii, and resulting μ_m of each ion were chosen randomly, by assuming that a 298 K neutral gas was ionized with uniform probability in a cylindrical region of radius 10 μm coaxial with $\hat{z}$, with B_2 = -5000 T/m^2 on. The ionization was done with an initial trapping voltage of 1.5 mV and the voltage was subsequently raised adiabatically to V_0 linearly in 0.1 s. Ions with higher values of μ_m were thus preferentially detrapped due to the cancellation of part of the low electrostatic trapping potential by the magnetic bottle term, constraining the range of allowed axial frequencies (through Eq. [4.4]), keeping all ion signals within the detection bandwidth used (205 Hz). In addition, the subsequent

increase in trapping depth to V_0 compressed the axial orbits to have $|z_{max}| < z_0/2$, thereby avoiding the regions of the cell where anharmonicity would be induced by deviations in the form of the trapping fields. To increase the signal amplitude, the axial motion was excited by a 1 V, 5 μsec voltage pulse, which would have excited an ion with zero axial energy to a 0.1 cm amplitude; this pulse was still weak enough that ions were not excited to the anharmonic regions of the cell. To improve the signal-to-noise ratio, 100 shots were simulated for each point in ω. The number of ions per shot has Poisson statistics with an average of ten ions remaining trapped per shot. This is 1.3 % of the ions formed, the others being in orbits that collide with the ion trap surfaces. New ions were chosen for each shot, but, if desired, the same ions could be used for successive repetitions.

The stimulated spin transitions were induced by a microwave field with $B_1 = 0.1$ G; this can be achieved with microwave power of 10 W, easily attained for the electron resonance frequency in a 1 T magnetic field (≅ 30 GHz). The field is assumed to be on during $t_1 = 1.79$ μsec, which gives unity probability for a spin transition at resonance. The spectrum of a single ESR line 2 MHz off resonance was simulated by sweeping the microwave frequency over 40 points in 250 kHz steps, giving a 10 MHz bandwidth.

Fig. 4.3 is a simulation of the signals including noise. Axial detection in each period took place over $\tau_{max} = 10$ sec, giving 0.1 Hz resolution. The signals were mixed with the reference frequency $\omega_r/2\pi = 23748.3$ Hz. The relationship between single-ion signal and axial amplitude is $I^z_{rms} = \frac{q\kappa}{2z_0}\omega_z\left\langle z^2\right\rangle^{1/2}$.[2] With a detector input impedance of 10^8 Ω, the axial energy decay rate due to the measurement is a negligible $\gamma_z/2\pi = 17$ mHz. White noise equal to the Johnson noise of a 10^8 Ω resistance at 4.2 K was added to each record.

The frequency domain signal from a single detection period is shown in Fig. 4.3(a). The axial frequencies of the 12 ions present in this shot are indicated by arrows. Note that the phases are random and the sensitivity at this point is marginal for single ions with the

assumed level of electrical noise. As outlined in the previous section, all "before" signals were processed by first being Fourier transformed. Once in the frequency domain, the original spectrum (e.g., Fig. 4.3(a)) was then "blanked" by applying a threshold function, equal here to three standard deviations above the mean intensity of the spectrum. All peaks above this line are preserved, while the points with intensities below the threshold are set to zero, as in Fig. 4.3(b); note that, in this example, 9 of the original 12 ion peaks survived the procedure. The blanked spectrum was then transformed back to the time domain, and was subsequently multiplied by the time-reversed "after" signal. In Fig. 4.3(c) the transforms of the time-reversed products are shown for 5 of the 40 values of ω simulated. Fig. 4.3(d) shows the resulting ESR spectrum, which reveals the simulated 2 MHz offset line with adequate signal-to-noise ratio $\cong$ 4. The total experimental time corresponding to the simulation is $\cong$ 24 hours.

4.1.d Discussion

If the electron experiences hyperfine coupling to nuclear spins in the ion, then there will be multiple resonance frequencies $\Delta\omega$. Spin-rotation and spin-orbit coupling will introduce additional structure.[10] Since the electron Larmor frequency can be made much higher than these couplings, the analysis generalizes: $S(\omega)$ is the autocorrelation function of the off-diagonal density matrix elements excited by the microwave pulses, mapped out by incrementing ω and repeating the experiment. The number of bits needed at each value of ω and the number of different values of ω will depend on the complexity of the spectrum.

The only broadening mechanisms relevant are those that lead to time dependence of a single ion's orbit during one shot. Aside from loss of energy to the detection circuit which leads to a minor damping of the axial amplitude, the other general linewidth mechanism is collisions with neutrals or other ions. In the absence of a detailed theory, it is simply noted that the latter can be eliminated by working, if necessary, with a single ion

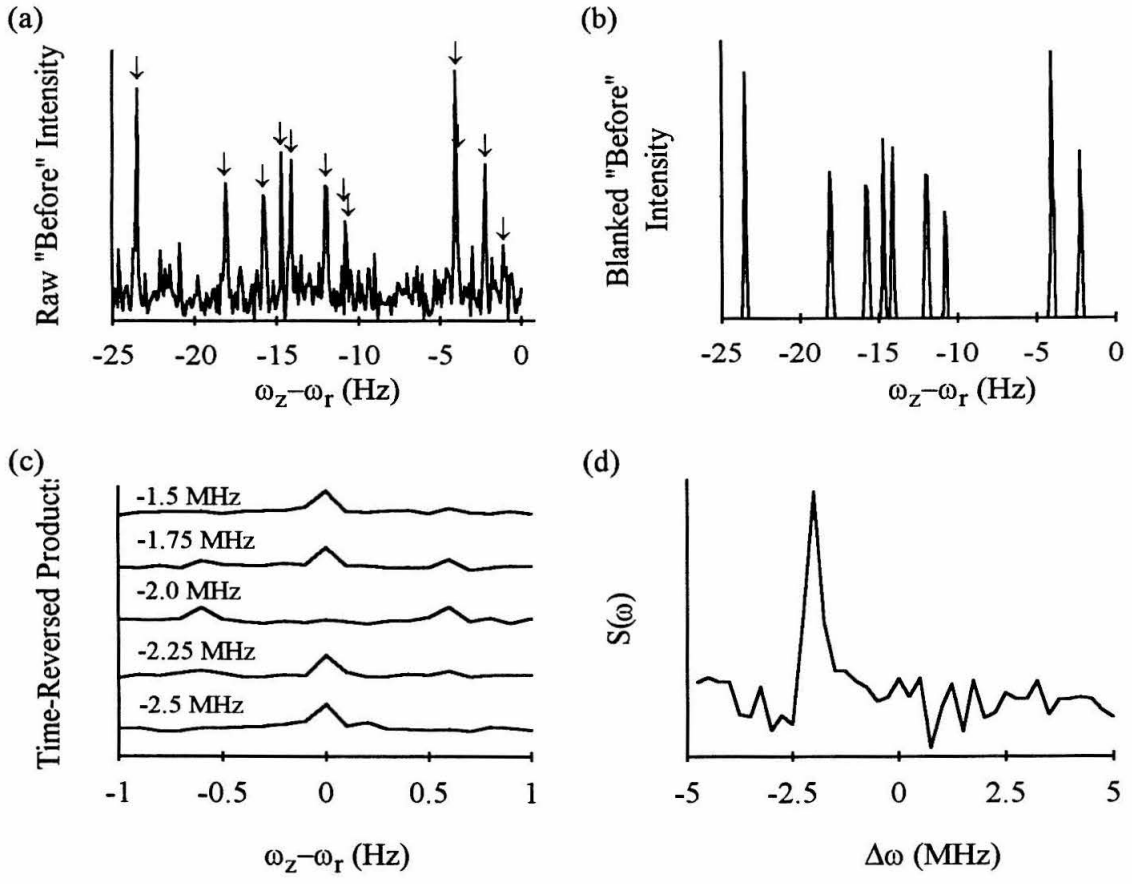


Fig. 4.3. Results of numerical simulation outlined in the text. (a) Fourier transform of signal from a single shot. The arrows indicate the 12 ions present. (b) The same transform after "blanking." (c) Time-reversed products for five frequencies with resonance offsets as labeled. (d) The ESR signal S(ω).

in the cell at one time and the former by working under collisionless conditions. In ICR, non-reactive, ion-neutral interactions are described by the reduced collision frequency,[11] and made negligible (< 1 Hz) at room temperature by operating at pressures below 10^{-8} torr.

Aside from the phenomenal sensitivity increase, the method as proposed is capable of recording single-ion spectra at resolution limited only by experimenter patience.

Resolution for spin-related Hamiltonian parameters will easily surpass that of existing approaches for highly reactive species: matrix-isolation studies ($\geq$ 100 kHz)[10] or hyperfine analysis of high-energy transitions ($\geq$ 1 MHz).[12] This will allow unprecedented high precision measurements of the many quantities which will determine the line frequencies: magnetic moments, hyperfine splittings, spin-rotation coupling, tunneling splittings, and electric field gradients.

The technique proposed may also be used for high-sensitivity nuclear magnetic resonance (NMR) spectroscopy of any sample in any phase that can be obtained with strong nuclear spin ordering and converted to trapped ions. Modulation of this order by NMR will appear as modulation of the hyperfine multiplet intensities in the ESR of ions produced and detected within the nuclear spin-lattice relaxation time. In addition, one may also propose schemes which employ spin-dependent work on the trapped ion motions to produce larger frequency shifts than those obtained in this section. Techniques involving spin-dependent work are proposed for both ESR (in the remainder of this chapter) and NMR (in Chapter 5).

4.2 *I*on *o*rbit *n*udging *t*hrough *r*esonance *i*nducing *c*yclotron *k*inematics: an IONTRICK for spin-dependent work

In the previous section, the existence of spin-dependent axial potentials allowed the development of a novel magnetic resonance spectroscopy of ions, where spin flips would be detected as small shifts in the monitored axial frequency. A single spin flip was used to incorporate ESR information. The procedure proposed in this section will employ additional spin flips to magnify the spin dependence of the ion motions. The extra flips will perform spin-dependent work on the axial mode, and the corresponding energy change will be detected via its shift of the cyclotron frequency in the presence of the magnetic bottle.[13] Since manipulation of the ion orbit in a spin-dependent fashion will be

made to yield shifts in the observed cyclotron spectrum, the technique has been named "*i*on *o*rbit *n*udging *t*hrough *r*esonance *i*nducing *c*yclotron *k*inematics" (IONTRICK).

For a given amount of axial energy, the amplitude of the axial oscillation contains information on the spin state of an ion. It will be shown that spin flips synchronized with the axial oscillation of an ion change the axial amplitude and thus serve as a test of the original spin state. The sign of the change depends on the original spin state, analogous to the situation in a Stern-Gerlach experiment. The magnitude of the change can be magnified by using multiple flips. Even so, the change in axial amplitude will usually be small and impractical to detect by direct observation of the axial motion. However, because of the coupling of the cyclotron motion to the axial motion mediated by a strong magnetic bottle, the amplitude change can be detected as a shift in the cyclotron frequency. Examination of Eq. [3.26a] shows that an increase in axial amplitude (*i.e.*, an increase in the axial quantum number k) should produce an increase in the cyclotron frequency for a positively charged ion.

4.2.a Axially synchronized spin flip cycles

The mechanism proposed uses the difference in the curvatures of U_+ and U_- as an energy pump or drain, depending on the initial spin state. Spin flips take place via π pulses at two different frequencies, corresponding to the resonances connecting the potentials at $z \approx 0$ and $z \approx \pm z_1$. Since feasible π pulse times for electrons are as low as $\approx$ 10 nsec, the pulses are considered instantaneous on the axial timescale. At $z = 0$, the resonance frequency is just the Larmor frequency ω_{L0}. At $z = \pm z_1$, however, the frequency ω_{L1} is given by (using Eq. [2.42] for the spin-dependent axial potentials, accurate to first order in the magnetic bottle field strength)

$$\omega_{L1} = \frac{1}{\hbar}\left(U_+(z = z_1) - U_-(z = z_1)\right) = \omega_{L0} - \frac{g|\mu_B|B_2}{\hbar} z_1^2 + O\left(B_2^2 z_1^4\right). \quad [4.23]$$

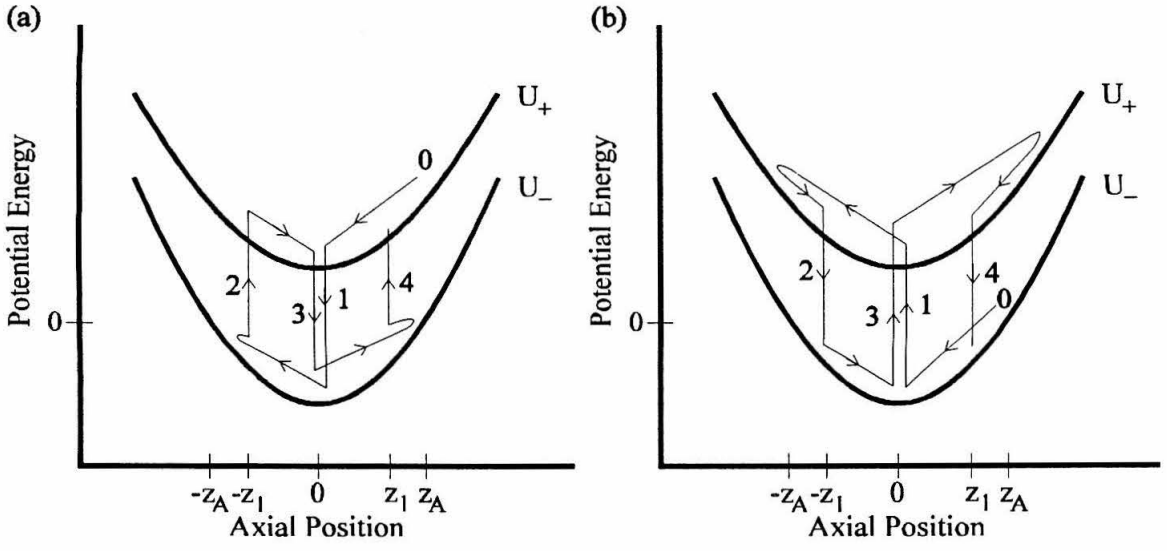


Fig. 4.4. Effects of one axial cycle with synchronized spin flips. π pulses of frequency ω_{L0} flip the spin as the ion goes through z=0 (steps 1 and 3), while π pulses of frequency ω_{L1} flip the spin as the ion moves toward the origin through $z=\pm z_1$. There is a net energy change after a full cycle, but spin is conserved. (a) Spin is up initially. (b) Spin is down initially. The dimensions are not to scale; differences in the curvatures of the two potentials are greatly exaggerated.

It is assumed initially, for simplicity, that the axial motion has been previously excited by a voltage pulse to the trapping electrodes to a reasonably large initial amplitude, z_A (.05 to .5 cm), so that the distribution of axial velocities is small compared to the mean axial velocity. Thus there will be a definite phase of the axial motion to which the magnetic resonance pulses will be synchronized. The ω_{L1} spin flipping positions, $\pm z_1$, are chosen as large as possible, but smaller than z_A by at least 10%, to ensure that the axial amplitude is still greater than z_1 after a given spin flipping experiment.

To distinguish between the two possible initial spin states, the spin flip sequence is chosen so that it will decrease the axial energy if the initial state is $|\alpha\rangle$, and increase the axial energy if the initial state is $|\beta\rangle$. A total of four π pulses are given in each axial period, as shown in Figs. 4.4. The ion is assumed to be at $z \approx z_A$ at time zero, and moving

toward the origin. Consider first the case where the initial spin state is $|\alpha\rangle$, as in Fig 4.4a. The ion experiences a π pulse (frequency ω_{L0}) as it passes through the origin, and moves to the U_ potential, losing energy $\hbar\omega_{L0}$ in the process (step 1 in Fig. 4.4a). Having gone from a shallower curve to a steeper curve, the ion turns at some point closer to the origin than $-z_A$, and then undergoes a spin flip (π pulse of frequency ω_{L1}) as it moves through $-z_1$, gaining energy $\hbar\omega_{L1}$ from the transition (step 2 in Fig. 4.4a). It is this step that sets an upper bound for z_1, since the turning point of the ion while in the U_ potential must always be farther from the origin than $\pm z_1$. Steps 3 and 4 of Fig. 4.4a are similar to steps 1 and 2, respectively. At the end of one axial period, the ion has lost an amount of energy equal to $2\hbar(\omega_{L0}-\omega_{L1})$. If this is repeated over N such cycles, the axial amplitude decreases to a value z_{N+} such that

$$U_+(z_{N+}) = U_+(z_A) - 2N\hbar(\omega_{L0}-\omega_{L1}). \qquad [4.24]$$

The explicit solution of this equation is

$$z_{N+} = \sqrt{z_A^2 - \frac{2N\hbar(\omega_{L0}-\omega_{L1})}{\frac{qV_0}{2d^2}\left(\mu_m - \frac{g}{2}|\mu_B|\right)B_2}}. \qquad [4.25]$$

The energy gaining case, where the initial spin state is $|\beta\rangle$, is analogous to the situation just described. The ion starts at $z \approx z_A$ at time zero in the spin down potential. As it moves through the origin, a π pulse of frequency ω_{L0} sends the ion to the spin up potential, with energy gain $\hbar\omega_{L0}$ (step 1 in Fig. 4.4b). The ion turns at a point beyond $-z_A$, since it has gone from a deeper to a shallower curve, then returns toward the origin. A π pulse of frequency ω_{L1} flips the spin as the ion moves through $-z1$, with energy loss $\hbar\omega_{L1}$ (step 2 in Fig. 4.4b). Steps 3 and 4 of Fig. 4.4b are analogous to steps 1 and 2, respectively. In contrast to the preceding case, the ion has gained a total amount of energy equal to $2\hbar(\omega_{L0}-\omega_{L1})$ after one axial period. If N such cycles are performed, the axial amplitude increases to a value z_{N-} such that

$$U_-(z_{N-}) = U_-(z_A) + 2N\hbar(\omega_{L0} - \omega_{L1}). \qquad [4.26]$$

Solving this equation,

$$z_{N-} = \sqrt{z_A^2 + \frac{2N\hbar(\omega_{L0} - \omega_{L1})}{\frac{qV_0}{2d^2}\left(\mu_m + \frac{g}{2}|\mu_B|\right)B_2}}. \qquad [4.27]$$

The axial amplitude changes predicted here are substantiated by exact numerical simulations of the full classical motion. A version of TRAJ.FOR designed to flip the spin instantaneously at z=0 and $z=\pm z_1$ at the times indicated above, and in Fig. 4.4, with axially synchronized spin flip cycles, shows axial amplitude changes that match Eqs. [4.25] and [4.27]; sample results are presented in Table [4.3] This agreement increases confidence in the approximations used to justify an expression for the axial potential good to first order in the magnetic bottle strength B_2 and coupled to the transverse motions only through the mechanical magnetic moment μ_m. The axial motion can indeed be considered decoupled from the transverse motions. The simple physical picture of two axial potentials differing only because of the spin state of the ion is valid.

As stated previously, these changes in axial amplitude will be monitored via shifts in the cyclotron frequency. This can be understood qualitatively by supposing that the cyclotron frequency is proportional to the average magnetic field experienced over an axial period. Greater axial amplitude carries the ion into the region where the magnetic bottle contribution B_2z^2 is more important. More precisely, the spin-dependent shifts due to the magnetic bottle will be shifted further by the spin-dependent change in the axial quantum number k. Squaring the quantum operator for z (Eq. [3.5a]) and taking its expectation value in a pure state $|nkls\rangle$, a quantum analogue of the square of the axial amplitude is obtained. Rearranging this and solving for k,

$$k = \frac{m\omega_z\langle z^2\rangle}{2\hbar} - \frac{1}{2}. \qquad [4.28]$$

Parameter	Run 1	Run 2	Run 3	Run 4
Original spin orientation	$\|\alpha\rangle$	$\|\beta\rangle$	$\|\beta\rangle$	$\|\beta\rangle$
N	109	109	10000	10000
ρ_+ (cm)	0.5	0.5	0.05	0.5
Trajectory integration time step (s)	5×10^{-9}	5×10^{-9}	6.9×10^{-8}	6.9×10^{-8}
Trajectory integration resolution (Hz)	500	500	10	10
$z_{N\pm}-z_A$ (μm) expected from Eqs. [4.25] and [4.27]	-0.78	0.78	73.7	71.2
$z_{N\pm}-z_A$ (μm) observed in trajectory integration	-0.69	0.69	72.3	61.2

Table 4.3. Axial amplitude changes due to axially synchronized spin flip cycles: the changes predicted via Eqs. [4.25] and [4.27] are compared to results from numerical trajectory simulations, produced by a version of TRAJ.FOR (see Appendix A). Simulation parameters are listed in the table. In all runs, m = 100 amu, q = +1, B_0 = 1 T, V_0 = 10 V, d = 1 cm, B_2 = 1000 T/m^2, z_A = 0.5 cm, z_1 = 0.4 cm, ρ_- = 0, and the ion is assumed to have the magnetic moment of an unpaired electron spin.

Assuming that the quantum numbers n and l for transverse motion remain constant, the observed shift after a spin flipping experiment is

$$\delta = \Delta(\Delta\omega_+) = \Delta\omega_+\left[k(z_N)\right] - \Delta\omega_+\left[k(z_A)\right]. \qquad [4.29]$$

Examining Eq. [3.26a] for the perturbative shift of the cyclotron frequency due to the magnetic bottle, and assuming $qB_2 > 0$, it is determined that $\Delta\omega_+$ increases with increasing k. Thus, the energy gaining experiment, where the initial state is $|\beta\rangle$, produces a positive frequency shift, while the energy losing case, with initial state $|\alpha\rangle$, leads to a negative shift.

Inserting Eqs. [3.26a] and [4.28], and remembering that $\omega_+ \gg \omega_-$,

$$\delta_\pm \approx \frac{qB_2}{2m}\left(z_{N\pm}^2 - z_A^2\right) \qquad [4.30]$$

for initial spin $m_s = \pm 1/2$. The 1/m dependence, which comes from the usual sensitivity of cyclotron frequency to mass, means that heavier ions require a greater number of spin flip cycles to separate spin states by a given cyclotron frequency difference. For a fixed trap size (d), the trapping voltage (V_0) may be adjusted for heavy ions to compensate for the mass effect. Inspection of Eqs. [4.25] and [4.27] reveals that the axial amplitude change depends on $1/\sqrt{V_0}$, in the limit of a small magnetic bottle and/or small mechanical magnetic moment; specifically, this is the limit of $\omega_z^2 \gg \mu_m B_2/m$. Expressing δ in terms of the fundamental experimental parameters, for initial spin $m_s = \pm 1/2$,

$$\delta_\pm = \mp\left(\frac{\omega_+}{\omega_+ - \omega_-}\right)\frac{N\hbar qB_2(\omega_{L0} - \omega_{L1})}{m\left[\frac{qV_0}{2d^2} + (\mu_m \mp |\mu_B|)B_2\right]}; \qquad [4.31a]$$

$$\delta_\pm = \mp\left(\frac{\omega_+}{\omega_+ - \omega_-}\right)\frac{Nqg|\mu_B|B_2^2 z_1^2}{m\left[\frac{qV_0}{2d^2} + (\mu_m \mp |\mu_B|)B_2\right]}. \qquad [4.31b]$$

This suggests that, with a limited frequency resolution in the detection mechanism, the applicability of the spin flipping technique may be extended to larger ions by lowering the trapping potential instead of resorting to very long trains of π pulses, in the limit of a small bottle. Estimating the practical limit of this process requires a thorough exploration of ion

cooling and trapping with low potentials. In the following, values of V_0 near those of known trapping experiments will be assumed.

4.2.b Calculated shifts for typical experiments

The inverse mass dependence of the axially synchronized spin flip effect (Eq. [4.30]) makes the choice of a small ion appropriate for the first experiments. The smallest molecular ion, H_2^+, has one unpaired electron and is therefore a good test ion. Typical numbers for the detection of its electronic spin state via ICR follow. However, with future experiments in mind, numbers are also presented for a spin 1/2 ion of 100 amu mass. The Penning trap considered has a characteristic dimension of 1 cm, with B_0 = 1 T, and a strong magnetic bottle, with B_2 = 1000 T/m^2. The cyclotron radius and the axial amplitude are both assumed excited to 0.5 cm. Under these conditions, the cyclotron frequency (in the absence of the bottle) of H_2^+ is approximately 7.6 MHz, and that of the 100 amu ion is 150 kHz.

With a trapping voltage of 10 V, 10 axially synchronized spin flip cycles give a cyclotron shift of ±1 Hz; thus, spin up and spin down are separated by 2 Hz. The voltage dependence discussed above may be put to experimental advantage by reducing V_0 to 5 V, giving the same shift with 8 cycles. At a trapping voltage of 10 V, the 100 amu ion requires 180 cycles to give a ±1 Hz shift. Reducing V_0 to 5 V decreases the number of cycles to 93; reducing V_0 further to 1 V decreases the number of cycles to 24.

4.2.c Proposed IONTRICK experimental sequence

Cyclotron frequency differences of the magnitude predicted above would be readily observed in ordinary ICR. However, in the presence of the bottle needed to couple spin and space variables, the distribution of thermal energies would give rise to distributions of cyclotron frequencies which would obscure the spin effects. This can be seen by substituting into Eq. [3.26a] values of the quantum numbers n, k, and l typical of

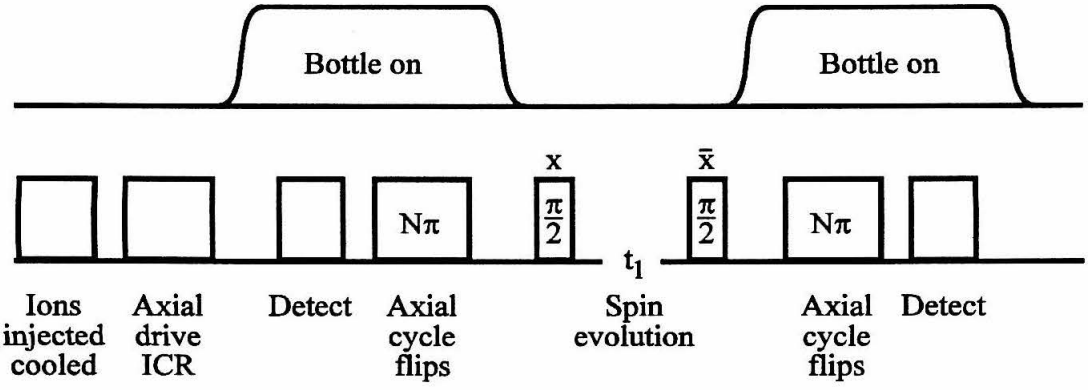


Fig. 4.5. Experimental sequence for magnetic resonance by ion cyclotron resonance via IONTRICK.

ion ensembles at ambient temperatures. At 300 K this effect gives an inhomogeneous distribution of the cyclotron frequency equal to ≈15% of its center value. The small shifts $\delta_{\pm}$ must be measured despite this distribution. This can be achieved via measurements of the cyclotron frequency taken before and after a magnetic resonance evolution period in which the bottle field is absent. Suppose the cyclotron frequency of an ion is measured just before applying N spin-flipping axial cycles. Let the initial measured frequency be $\omega_+(t=0)$ and let the time after the N cycles be t_N. Then, $\omega_+(t=t_N)$ is $\omega_+(t=0)+\delta_{\pm}$ for $m_s = \pm 1/2$. After the application of another set of N cycles, ending at time t_{2N}, $\omega_+(t=t_{2N})$ is $\omega_+(t=0)+2\delta_{\pm}$ for $m_s = \pm 1/2$. However, if some event caused a spin flip between the first and the second set of N cycles, $\omega_+(t=t_{2N})$ would be $\omega_+(t=0)$ for both spins. In the following, it is assumed that the only significant mechanism of spin flips in the period between the π trains is Larmor frequency irradiation. Spin-lattice relaxation is assumed negligible, since no effective mechanism is present for small molecules under collisionless conditions.

ESR information is encoded by providing a variable mechanism for spin reorientation between the two sets of N spin-flipping axial cycles. After the first set of N

cycles, the bottle field is turned off (to make the magnetic resonance spectrum independent of the translational quantum numbers), and a $\pi/2$ pulse is applied, followed by another one at a variable time t_1 later. The bottle is then turned on again, the second set of N cycles is applied, and the cyclotron frequency is measured. As t_1 is varied from one run to the next in multiples of the axial period $2\pi/\omega_z$, the probability of finding the ion in one spin state or the other oscillates, and ESR is observed via the cyclotron shifts. Specifically, each ion exhibits one of three outcomes: a shift by $2\delta_+$, a shift by $2\delta_-$, or no shift. If $\Delta\omega$ is the resonance offset, the probability of the first two outcomes is $\frac{1}{4}\left[1+\cos(\Delta\omega t_1)\right]$ each, and that of the third is $\frac{1}{2}\left[1-\cos(\Delta\omega t_1)\right]$. The signal $S(t_1)$ consists of one bit (a zero or a one) placed in each of these three "channels." To obtain an accuracy of at least M bits in each channel, the single-ion experiment must be repeated 4M times for each value of t_1. The number of bits needed and the number of points in t_1 will depend on the complexity of the spectrum $S(\omega_1)$, the Fourier transform of $S(t_1)$.

If the experiment is performed on a single ion, the same ion may be used again as t_1 is varied, or a new ion of the same species may be trapped. Because of the probabilistic nature of the spin state transitions, however, it will save time to perform this experiment on small clouds of ions. Since the distribution of cyclotron frequencies $\omega_+(t=0)$ is orders of magnitude greater than the shifts $\delta_\pm$, there is negligible probability of the signal from two ions falling in overlapping ranges. Thus the limit on the number of ions will likely be determined by ion-ion interactions, a subject of future study.

A schematic timing diagram for the experiment is presented in Fig. 4.5. It includes an initial period of axial and ICR excitation; this is advantageous, since larger axial and cyclotron amplitudes increase both the strength of the signal observed, and the effects of the magnetic bottle. It may also be necessary to repeat the ICR excitation pulse before every run, since the cyclotron frequency detection circuit will damp the cyclotron motion considerably, as discussed below.

4.3 Experimental considerations

A variety of experimental issues has been addressed in order to quantify the technical challenges posed by the proposals presented in this chapter. The effect of microwave pulses on trapped ion motions will be shown to be insignificant. The application of microwave power for ESR pulses, however, will be enhanced by using the normal modes of cylindrical microwave cavity operated as a Penning trap. A novel tool, selective detrapping of ions by a magnetic bottle in the presence of a weak axial trapping potential, will be developed. A dechirping algorithm to extract sharp cyclotron frequency information in spite of continuous chirping of the signal during the detection process will be derived.

4.3.a Cylindrical Penning trap and microwave cavity

The hyperbolic Penning trap described above provides a highly precise electrostatic field by virtue of its design. Further precision is possible through the use of small compensation electrodes (see Fig. 1), and careful design of trap geometry enables compensation without distortion of ω_z.[14] However, this type of trap requires delicate machining, and is thus exceedingly difficult to build. A cylindrical Penning trap, a simple cylindrical ring with flat endplates, is a better choice for the proposed experiment.[15] With orthogonal anharmonicity compensation, the trap is suitable for high precision work. Furthermore, this trap forms a microwave cavity with well-known properties,[2] an important consideration in achieving electron resonance. Final design of the actual trap used will involve the numerical solution of Laplace's equations to find an adequate choice of parameters providing stable and precise trapping over a large volume (up to $\approx$.5 cm radius). The boundary conditions will be complicated by the various necessary imperfections of the trap. Several holes will be needed for ion injection, vacuum,

microwave irradiation, etc. In addition, the cylindrical ring will be split into four quadrants; thus, the trapping ring will also excite ICR and detect cyclotron motion.

Several issues related to the design of a combination microwave cavity and Penning trap are discussed below. These include two approaches to studying the effect of microwave pulses (for ESR) on the trapped ion motions. The first of these uses the direct approach of trajectory simulations; the second approximates the pulses via the ponderomotive potential.[16] In order to understand the form of this potential, the structure of the normal modes of a cylindrical microwave cavity will be quantified, and particular modes will be chosen for the proposed experiments.

4.3.a (i) Microwave pulse effect on ion motion studied via full trajectory simulations

The first approach used to investigate the consequences of applying strong microwave pulses to trapped ions employed a trajectory simulation based on the numerical integration program TRAJ.FOR (see Appendix A). This version added the magnetic and electric fields due to microwave radiation to the standard Penning trap electromagnetic configuration (this was done in the DERIVS subroutine of TRAJ.FOR). Simulations with this program showed a negligible microwave contribution to the ion motion.

The microwave field, composed of the magnetic component $\vec{B}_1$ and the electric component $\vec{E}_1$, is described by the equations

$$\vec{B}_1 = B_1\left[\hat{x}\cos(\omega t) + \hat{y}\sin(\omega t)\right] \qquad [4.32]$$

and

$$\vec{E}_1 = E_1\left[\hat{x}\sin(\omega t) - \hat{y}\cos(\omega t)\right]. \qquad [4.33]$$

Parameter	Run 1	Run 2	Run 3
Simulation length (s)	1.84×10^{-8}	1.84×10^{-8}	1.78×10^{-6}
ρ_+ (cm)	0.05	0.5	0.5
Δx = final x (x_{on}) with microwaves minus final x (x_{off}) without microwaves (m)	3.0×10^{-8}	3.0×10^{-8}	1.7×10^{-8}
$\Delta x / x_{off}$	6×10^{-5}	6×10^{-6}	-6×10^{-5}
Δy = final y (y_{on}) with microwaves minus final y (y_{off}) without microwaves (m)	-2.8×10^{-10}	-2.8×10^{-10}	-2.0×10^{-8}
$\Delta y / y_{off}$	3×10^{-5}	3×10^{-6}	4×10^{-6}
Δz = final z (z_{on}) with microwaves minus final z (z_{off}) without microwaves (m)	-4.3×10^{-15}	-4.4×10^{-14}	-3.0×10^{-10}
$\Delta z / z_{off}$	-9×10^{-13}	-9×10^{-12}	-7×10^{-8}

Table 4.4. Translational coordinate changes due to microwave pulses with ω = 28 GHz and B_1 = 10 G, simulated with a version of TRAJ.FOR (see Appendix A). Simulation parameters are listed in the table. In all runs, m = 100 amu, q = +1, B_0 = 1 T, V_0 = 10 V, d = 1 cm, B_2 = 1000 T/m^2, z_A = 0.5 cm, ρ_- = 0; the ion is assumed to have the magnetic moment of an unpaired electron spin, and the integration time step is 1×10^{-12} s.

The experimentally reasonable value B_1 = 10 G, which gives a π pulse time of 17.8 nsec, implies a corresponding electric field strength E_1 = 3 kV/cm. Assuming the ion under study to be a 100 amu paramagnetic ion (with an unpaired electron spin supplying the

magnetic moment), the Larmor frequency is $\omega_{L0}/2\pi$ = 28.01 GHz at z = 0 and $\omega_{L1}/2\pi$ = 27.57 GHz at z_1 = 0.4 cm. The integration time step of 1×10^{-12} s thus sampled the ion motion 35 times per microwave period. Simulations were performed over lengths of time equal to both 1 pulse and 100 pulses; the results of runs with and without the microwave field were compared.

The results, presented in Table 4.4, show translational changes that are negligible compared to the axial amplitude change expected from the axially synchronized spin flip sequence. For example, the simulation with 100 pulses showed an axial coordinate decrease of 4×10^{-12} m. A 100 pulse sequence is equivalent to 25 axially synchronized spin flip cycles, which, under conditions similar to those in Table 4.3, would change the axial amplitude by 2×10^{-7} m. This is larger than the microwave pulse effect by a factor of 50000, suggesting that the effect of microwave radiation of trapped ion motions is negligible.

4.3.a (ii) Microwave pulse effect on ion motion studied with the ponderomotive potential for cylindrical cavity modes

While the simulations presented in the previous section suggest that the microwave field necessary to stimulate ESR transitions does not perturb the ion motion appreciably, the results do not offer complete proof of this. An in-depth study via direct trajectory integration should include a study of convergence as a function of the time step used; it is likely that sampling each microwave period 35 times, as was done above, does not provide sufficiently accurate results. However, further work by this method would be time-consuming and limited by current computational power. The fact that the microwave frequency is much higher than all of the trapped ion frequencies, however, allows the introduction of an effective, time-independent "ponderomotive" potential.[16] With it, the ion motion under the effect of microwave irradiation may be simulated while still using the cyclotron period as the relevant sampling time scale.

Irradiation at a frequency ω much greater than all of the trapped ion motional frequencies ($\omega_+,\omega_-,\omega_z$) allows the use of the oscillation-center approximation.[16] Microwave radiation may then be accounted for via a ponderomotive potential Φ,

$$\Phi = \frac{q^2}{m^2\omega^2}\left\langle \frac{E_0^2}{2} \right\rangle, \qquad [4.34]$$

where E_0 is the time-averaged microwave field strength. The oscillation center r_0 of the ion experiences the acceleration

$$\frac{d^2 r_0}{dt^2} = -\nabla\Phi. \qquad [4.35]$$

Poole[17] gives equations for the standing magnetic and electric fields in a cylindrical microwave cavity corresponding to TM_{mnp} modes:

$$H_r = -mH_0\left[\frac{J_m(k_c r)}{k_c r}\right]\sin(m\phi)\cos(k_z z); \qquad [4.36a]$$

$$H_\phi = -H_0 J'_m(k_c r)\cos(m\phi)\cos(k_z z); \qquad [4.36b]$$

$$H_z = 0; \qquad [4.36c]$$

$$E_r = -\sqrt{\frac{\mu}{\varepsilon}}H_0\frac{k_z}{\sqrt{k_c^2+k_z^2}}J'_m(k_c r)\cos(m\phi)\sin(k_z z); \qquad [4.36d]$$

$$E_\phi = \sqrt{\frac{\mu}{\varepsilon}}H_0\frac{k_z}{\sqrt{k_c^2+k_z^2}}m\left[\frac{J_m(k_c r)}{k_c r}\right]\sin(m\phi)\sin(k_z z); \qquad [4.36e]$$

$$E_z = \sqrt{\frac{\mu}{\varepsilon}}H_0\frac{k_c}{\sqrt{k_c^2+k_z^2}}J_m(k_c r)\cos(m\phi)\cos(k_z z); \qquad [4.36f]$$

where H_0 is the maximum magnetic field strength in the cavity, J_m is the regular Bessel function of order m, J'_m is its derivative with respect to the argument. Furthermore,

$$k_c = \frac{(k_c a)mn}{a}, \qquad [4.37a]$$

$$k_z = \frac{p\pi}{d}, \qquad [4.37b]$$

with d the cavity length, a the cavity radius, and $(k_c a)mn$ the n^{th} root of the m^{th} order Bessel function $J_m(k_c r)$.

It is clear from the magnetic field equations that the TM_{021} mode is excellent for delivering π pulses at z = 0, while the TM_{022} mode provides power for pulses closer to the axial ends of the cavity. Restricting the discussion to m = 0,

$$\left\langle E_0^2 \right\rangle = E^2 = E_r^2 + E_\phi^2 + E_z^2; \qquad [4.38a]$$

$$E^2 = \frac{\mu}{\varepsilon}\frac{H_0^2}{k_c^2 + k_z^2}\left\{k_c^2\left[J_0(k_c r)\cos(k_z z)\right]^2 + k_z^2\left[J_1(k_c r)\sin(k_z z)\right]^2\right\}. \qquad [4.38b]$$

This result is achieved with the help of the Bessel identities

$$\frac{d}{dx}J_n(x) = \frac{J_{n-1}(x) - J_{n+1}(x)}{2} \qquad [4.39a]$$

and

$$J_{-n}(x) = (-1)^n J_n(x). \qquad [4.39b]$$

Evaluating the necessary partial derivatives,

$$\frac{\partial E^2}{\partial r} = \frac{\mu}{\varepsilon}\frac{H_0^2}{k_c^2 + k_z^2}\left\{J_0(k_c r)J_1(k_c r)\left[k_c k_z^2 \sin^2(k_z z) - 2k_c^3 \cos^3(k_z z)\right]\right.$$

$$\left. - k_c k_z^2 J_1(k_c r)J_2(k_c r)\sin^2(k_z z)\right\}; \qquad [4.40a]$$

$$\frac{\partial E^2}{\partial \phi} = 0; \qquad [4.40b]$$

$$\frac{\partial E^2}{\partial z} = \frac{\mu}{\varepsilon}\frac{H_0^2}{k_c^2 + k_z^2}\left\{k_z^3\left[J_1(k_c r)\right]^2 - k_c^2 k_z\left[J_0(k_c r)\right]^2\right\}\sin(2k_z z). \qquad [4.40c]$$

Before proceeding further, units will be examined, and H_0 will be replaced with the more experimentally suited quantities of power P (in watts) and loaded cavity quality factor Q. The derivative equations are in MKS units, so E is in volt/meter, lengths are in meters, and H_0 is in ampere/meter. Poole[17] gives an approximate expression relating the magnetic field strength H_1 (in G = oersted) in a cavity to P and Q:

$$\left\langle H_1^2 \right\rangle \approx 10^{-3} PQ\left(\frac{V_w}{V_c}\right), \qquad [4.41]$$

where V_w is the volume of 1 wavelength in the waveguide and V_c is the volume of the cavity. (This equation is used only as an estimate, since it is really for a TE_{01} waveguide only.) Since 1 amp/m = $4\pi\times10^{-3}$ oersted,

$$H_0[in\ amp/m] = H_1[in\ gauss = oersted] \times \left[\frac{\text{amp/m}}{4\pi \times 10^{-3}\ \text{oersted}}\right]. \qquad [4.42]$$

Combining the last two equations, and using the approximation $H_1 = \sqrt{\left\langle H_1^2 \right\rangle}$,

$$H_0 \approx \frac{1}{4\pi \times 10^{-3}} \sqrt{10^{-3} PQ\left(\frac{V_w}{V_c}\right)}. \qquad [4.43]$$

Using a cavity of 4 cm length and 1 cm radius, and approximating the waveguide cross-section as 1 cm^2 and the wavelength as 1 cm, the volume ratio is $V_w/V_c = 1/(4\pi)$, and the square of the magnetic field is $H_0^2 \approx 0.5PQ$.

This discussion now culminates with the calculation of the acceleration $\bar{a}$ contributed to the trapped ion motion by the ponderomotive potential. Combining the ponderomotive equations,

$$\bar{a} = \frac{-q^2}{2m^2\omega^2} \nabla \left\langle E^2 \right\rangle; \qquad [4.44]$$

this expression is then expanded, separated into its Cartesian components, and inserted into the DERIVS subroutine of a version of the TRAJ.FOR trajectory integration program (see Appendix A).

Several simulations were carried out with such a program, integrating the equations of motion for H_2^+ in a magnetic field of 1 T, with trapping potential V_0 = 1 V, characteristic trap dimension d = 1.5 cm, and no magnetic bottle. Motion was simulated over the course of 1 μs, with a sampling time of 10^{-11} sec, allowing the observation of a total of seven cyclotron periods and one quarter of an axial period ($\omega_+/2\pi$ = 7.6 MHz, $\omega_-/2\pi$ 3.5 kHz, and $\omega_z/2\pi$ = 232 kHz). Results for pulses applied with TM_{021} and TM_{022} modes were compared, using PQ values of zero and 10^5 W. The translational positions and velocities were converted into quantum numbers to extract the mode energies. The quantum numbers were found to remain constant in all cases, indicating that the microwave pulses have a negligible effect on the ion motions. This is easily understood in terms of the ponderomotive potential, since it scales inversely with the square of the applied frequency.

4.3.b Selective ion detrapping with the magnetic bottle

It is often necessary to control the spread in orbits over an ensemble in an ion trap. Most techniques for this involve cooling schemes or control of the initial neutral gas or the ionization conditions. An alternative method is presented in this section to selectively make the axial motion detrapped for ions with transverse orbits beyond a threshold value. This scheme, employed in the ESR proposal depicted in Fig. 4.1, operates by adjusting two independent parameters in the expression for the axial potential. One of these terms depends on the transverse orbit and can be used to make the axial potential repulsive for ions with too large a mechanical magnetic moment.

The expression for the axial frequency in the presence of a magnetic bottle, neglecting spin, can be written from Eq. [2.42] as

$$\omega_z = \sqrt{\frac{qV_0}{md^2} + \frac{2\mu_m B_2}{m}}. \qquad [4.45]$$

For bound axial motion, the ions oscillate harmonically with this frequency. However, if $\frac{qV_0}{md^2} + \frac{2\mu_m B_2}{m} \leq 0$, the motion is unbound, and ions are detrapped. Neutrals may be ionized with V_0 positive and B_2 negative such that ions will be detrapped immediately if

$$\mu_m \geq \frac{qV_0}{2d^2|B_2|}. \qquad [4.46]$$

Expanding the expression for the mechanical magnetic moment,

$$\omega_+\rho_+^2 + \omega_-\rho_-^2 \geq \frac{V_0}{d^2|B_2|}. \qquad [4.47]$$

As a numerical example, consider a singly charged positive ion of mass 100 amu in a Penning trap with characteristic dimension d = 1 cm, a magnetic bottle strength $B_2 = -1000$ T/m^2, and a 1 T magnetic field. Ionization under a trapping voltage V_0 = 1 V produces axially bound ions only for cyclotron radii $\rho_+ \leq 1.8$ mm, with the equal sign being the limit of zero magnetron radius ρ_-.

4.3.c Signal detection

Detection circuits for the axial and cyclotron motions are shown in Fig. 4.6. The circuits are entirely analogous, since both detect the image currents produced on opposing plates by the motion of charge in the trap. The axial circuit (Fig. 4.6a) uses the trap endplates for driving and detecting the axial motion, while the cyclotron circuit (Fig. 4.6b) uses two opposing quadrants of the slit cylindrical ring. The drive voltage in each case is used for coherent excitation of the respective modes. The resistor damps the motion by removing energy from the observed oscillation.

4.3.c (i) Single-ion sensitivity

The proposals presented in this chapter rely on the electrical detection of trapped ion motions with single-ion sensitivity. Single molecular ions (*e.g.*, N_2^+) have been detected via their axial frequencies by Pritchard's group at MIT.[18] Their detection circuit

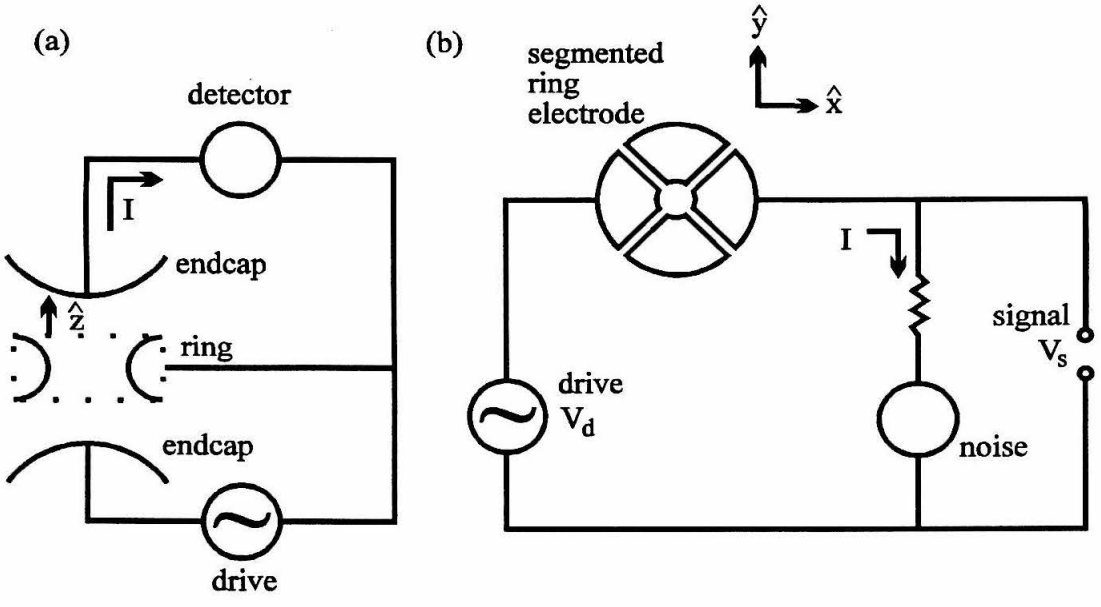


Fig. 4.6. Detection and driving circuits. (a) Axial circuit. (b) Cyclotron circuit. Adapted from Ref. 2.

consists of a superconducting tank and transformer stage leading into a superconducting quantum interference device (SQUID) detector. For such a circuit with quality factor Q, effective inductance L, resonance frequency ω, and detection bandwidth B, the time-domain signal-to-noise ratio is

$$\frac{S}{N} = I_{rms}\sqrt{\frac{\omega LQ}{4k_B TB}}, \qquad [4.48]$$

where T is the temperature of the circuit, k_B is Boltzmann's constant, and the rms current is given by

$$I_{rms} = \frac{\kappa}{2}\sqrt{\frac{q^3 V_0}{2m}}\frac{|z_A|}{dz_0}. \qquad [4.49]$$

The ion's axial amplitude is denoted by $|z_A|$, while d is the Penning trap characteristic dimension, z_0 is half the trap axial length, and κ is a constant of order unity. Combining

Parameter	Value cited in Ref. 19	Value proposed for ESR experiments
n_i	10	1
$\|z_A\|$ (cm)	0.02	0.25
$\omega_z/2\pi$ (kHz)	160	23.750
Q	12000	300
L (H)	5.7×10^{-3}	0.5
B (Hz)	13	79
time-domain S/N	5	0.1

Table 4.5. Extension of SQUID detection to more massive ions, with lower Q.

these two equations with the definition of the axial frequency, $\omega_z = \sqrt{qV_0/md^2}$, to reveal the main trends in the signal-to-noise ratio,

$$\frac{S}{N} \propto \frac{\omega^{3/2} L^{1/2} Q^{1/2}}{B^{1/2}} |z_A| n_i, \qquad [4.50]$$

where n_i is the number of ions contributing in phase at frequency ω. This allows a direct comparison to the MIT experiments, aiding the extension of the technique to more massive ions in the ESR proposals presented here. The comparison presented in Table 4.5 shows how the superconducting tank / SQUID circuit can be altered to give a time-domain signal-to-noise ratio of ≈ 0.1. Fourier transformation with resolution Δ can then be used to yield an effective signal-to-noise improvement of $(B/\Delta)^{1/2}$. Note that the ESR simulations presented in section 4.1.c have a similar time-domain S/N, illustrating the power of Fourier transformation in extracting signal information from time-domain noise.

4.3.c (ii) Cyclotron signal dechirping

The detection process removes quanta from the detected mode. If the detected frequency depends on the number of quanta in the mode, the frequency will be shifted continuously, or "chirped," throughout the detection period. This situation exists when the transverse trapped ion modes are detected in the presence of a magnetic bottle. Axial detection does not suffer from this shortcoming, since the bottle-perturbed axial frequency does not have any axial quantum dependence (Eq. [3.26c]). The cyclotron frequency, however, is linearly dependent on the cyclotron energy (Eq. [3.26a]). Since small cyclotron frequency shifts due to the axially synchronized spin flip experiment will be measured in the IONTRICK, any other mechanism which also shifts the frequency must be understood. Restricting the discussion to cyclotron effects only, and restating the classical expression for this phenomenon as derived in Chapter 2,

$$\Delta\omega_+ = -\frac{q}{2m}B_2\rho_c^2. \qquad [4.51]$$

Assuming that $qB_2 > 0$, a uniform decrease in the cyclotron radius leads to a uniform time-dependent increase (chirp) in the magnetic bottle shift of the cyclotron frequency. If this increase can be predicted accurately, its effect may be removed from the problem by a variety of methods (sweeping a reference frequency at the same rate, manipulating the Fourier transformation mathematically, etc.) which demodulate the chirp. This is very much like some problems encountered in radar observation. In this section an apparently novel dechirping algorithm is presented which is time-efficient by virtue of being based on the fast Fourier transform.

The damping constant γ_+ defines the rate at which the cyclotron energy decays:[2]

$$E_+(t) = E_+(t=0)e^{-\gamma_+ t}; \qquad [4.52]$$

$$\gamma_+ = \left(\frac{q\kappa'}{2\sqrt{2}d}\right)^2 \frac{R}{m}\frac{\omega_+}{\omega_+ - \omega_+}, \qquad [4.53]$$

where R is the resistance in ohms and κ' is a constant of order unity, exactly equal to one for perfectly flat endcaps. Since the cyclotron energy is proportional to the square of the cyclotron radius, and since the magnitude of the bottle shift to ω_+ is also proportional to the square of the cyclotron radius, the magnitude of the shift also decays with time constant γ_+. The cyclotron motion during detection has a time-dependent frequency. The small frequency shifts $\delta_\pm$ produced by the IONTRICK experiment can be calculated, since the peaks in the demodulated ICR Fourier transforms are just the frequency at the end of the pre-experiment detection period and the frequency at the beginning of the post-experiment detection period. For example, for the H_2^+ experimental values given above, the total cyclotron frequency shift due to damping during one second of detection is around 150 Hz. Algorithms will now be presented to remove this chirp and extract the cyclotron frequency at the beginning or the end of a detection period. These algorithms have been tested and have been shown to have the added attraction of being robust to noise.

Let the signal due to an ensemble of trapped ions be represented by

$$S(t) = \sum_j a_j e^{i\omega_{+j}(t)t}. \qquad [4.54]$$

Defining

$$\omega_{+j}(t) = \omega_+ + \Delta\omega_{+j}(t) = \omega_+ + \Delta\omega^0_{+j} e^{-\gamma_+ t} \qquad [4.55]$$

and establishing a reference frequency ω_r such that

$$\Delta\omega = \omega_+ - \omega_r, \qquad [4.56]$$

the signal becomes

$$S(t) = \sum_j a_j e^{it\left(\Delta\omega + \Delta\omega^0_{+j} e^{-\gamma_+ t}\right)}. \qquad [4.57]$$

The dechirping function used to extract the frequency at t=0 is defined by

$$f(t,\omega) = e^{it\left[\left(1-e^{-\gamma_+ t}\right)(\omega-\Delta\omega)\right]}; \qquad [4.58]$$

the desired dechirped signal $G(\omega)$ is then obtained via

$$G(\omega) = \int_0^\infty S(t) f(t,\omega) e^{-i\omega t} dt, \qquad [4.59]$$

the Fourier transform of the product of the signal and the dechirping function. The dechirping can be rewritten in terms of Fourier transforms to take advantage of the computational speed of the fast Fourier transform (FFT).[20] This also allows the separation of the problem into a larger part which is calculated once for each set of similar experiments (*e.g.*, with the same γ_+) and a smaller part involving actual individual signals. The dechirping procedure begins by calculating the matrix $F(\omega,\omega')$, which requires one FFT over ω' for each value of ω. This is calculated only once, since it remains constant for different signals as long as the damping constant γ_+ and the reference offset $\Delta\omega$ do not change:

$$F(\omega,\omega') = \int_0^\infty f(t,\omega) e^{-i\omega t} e^{i\omega' t} dt. \qquad [4.60]$$

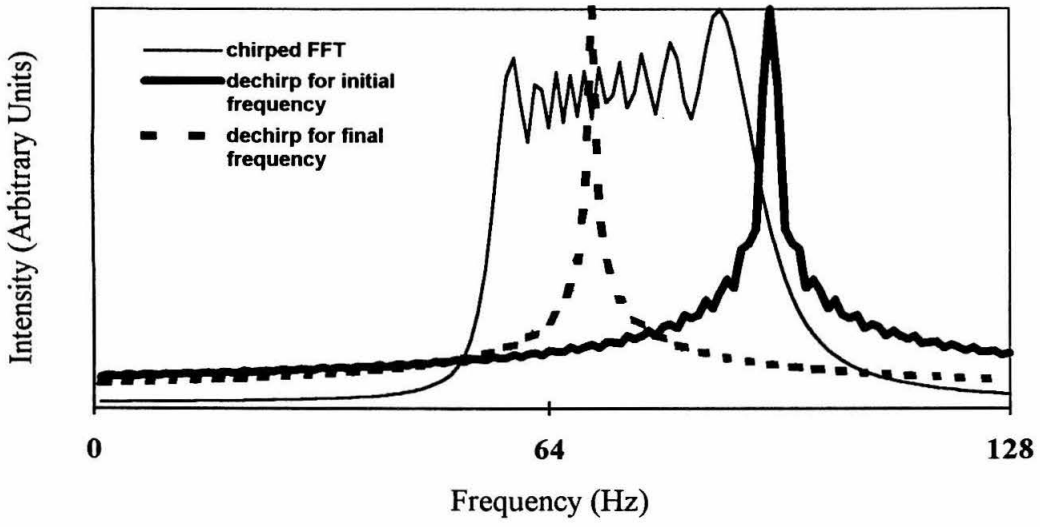
The FFT of each signal is then computed:

$$S(\omega) = \int_0^\infty S(t) e^{-i\omega t} dt. \qquad [4.61]$$

The dechirped signal is finally calculated by integrating over the product of the transforms calculated in the two previous equations:

$$G(\omega) = \int_{-\infty}^\infty S(\omega) F(\omega,\omega') d\omega'. \qquad [4.62]$$

(a)



(b)

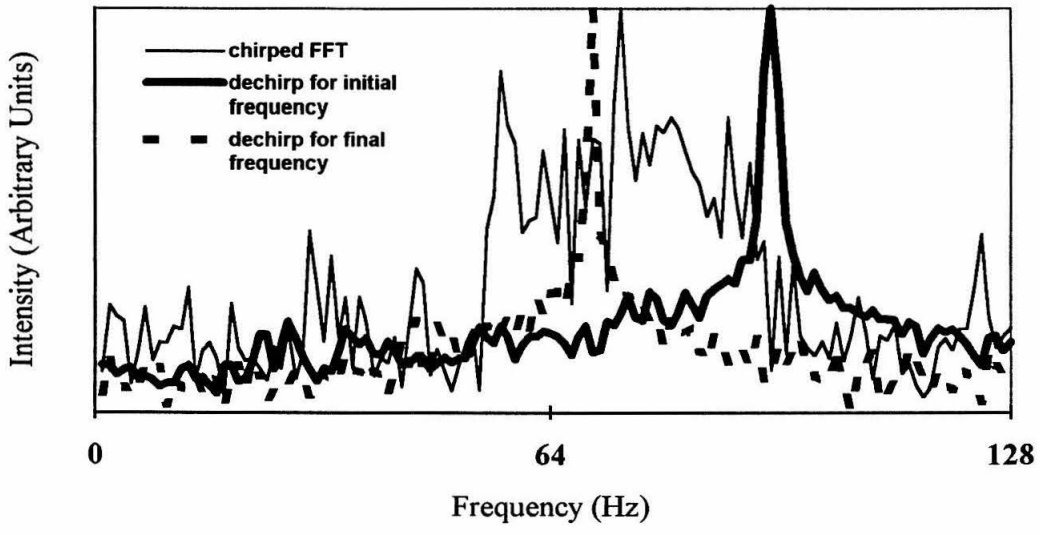


Fig. 4.7. Demonstration of dechirping for a single-ion spectrum (a) without and (b) with noise. Simulation parameters are described in the text.

The last two transforms, $S(\omega)$ and $G(\omega)$, are repeated for each signal over the course of an experiment. Note that the frequency extracted by the above procedure is that at $t = 0$;

this is set by the definition of f(t,ω). The signal frequency at t = τ may also be extracted by using the dechirping function

$$f(t,\omega) = e^{it\left[\left(1-e^{-\gamma_+(t-\tau)}\right)(\omega-\Delta\omega)\right]}. \qquad [4.63]$$

The frequencies extracted by these algorithms are the *instantaneous* frequencies $\omega_j(t)$, given by

$$\omega_j(t) = \frac{d}{dt}\left[\left(\Delta\omega + \Delta\omega^0_{+j}e^{-\gamma_+ t}\right)t\right] = \Delta\omega + \Delta\omega^0_{+j}(1-\gamma_+ t)e^{-\gamma_+ t}. \qquad [4.64]$$

Fig. 4.7 demonstrates the application of this technique to an ion with $\Delta\omega/2\pi$ = 55 Hz, $\Delta\omega^0_{+j}/2\pi$ = 40 Hz, and $\gamma_+/2\pi$ = 1 Hz using a 256-point simulated data set and 1 Hz resolution. Fig. 4.7(a) shows the successful dechirping of the spectrum in the absence of noise, isolating initial and final frequencies of 95 Hz and 70 Hz, respectively, as predicted from Eq. [4.64]. In Fig. 4.7(b), noise equal to the amplitude of a single ion's signal has been added without impairing the success of the dechirping algorithm. Fig. 4.8 demonstrates dechirping for a simulated signal from ten ions with random phases, including noise of amplitude equal to three times a single ion's signal. The ion parameters are tabulated in Table 4.6. This example illustrates the power of the dechirping technique: while no lines can be distinguished from noise in the chirped spectrum, dechirping reveals all ten peaks in both the initial-frequency- and final-frequency-dechirped spectra and allows the definite separation of most ion lines from the noise.

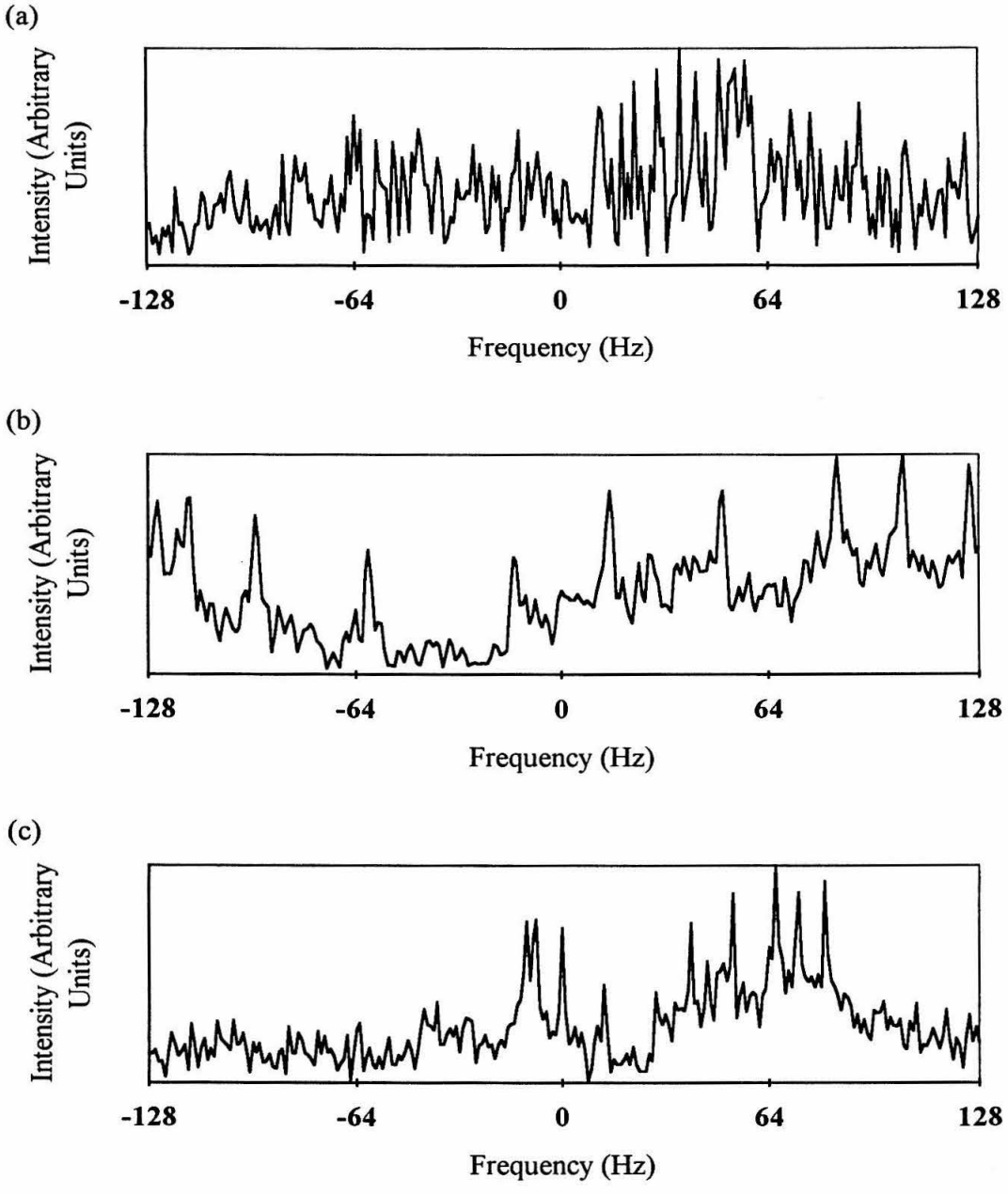


Fig. 4.8. Dechirping of a simulated ten-ion spectrum with noise amplitude equivalent to three ions. Parameters are described in the text and in Table 4.6 (a) Chirped spectrum. (b) Spectrum dechirped to reveal the initial frequencies. (c) Spectrum dechirped to isolate the final frequencies.

Ion # (index j)	$\omega^{0}_{+j}/2\pi$ (Hz)	phase (degree)	$\omega_j(t=0)/2\pi$ (Hz)	$\omega_j(t=\tau)/2\pi$ (Hz)
1	-180	0	-125	-11
2	-170	195	-115	-8
3	-150	70	-95	0
4	-115	290	-60	13
5	-70	20	-15	29
6	-40	120	15	40
7	-5	235	50	53
8	30	260	85	66
9	50	180	105	73
10	70	45	125	81

Table 4.6. Individual ion parameters for the ten-random-ion dechirping simulation presented in Fig. 4.8. The following parameters are equal for all ions: $\Delta\omega/2\pi$ = 55 Hz, $\gamma_+/2\pi$ = 1 Hz, resolution = 1 Hz, number of points = 256, τ = 1 sec.

4.3.d Vacuum and temperature requirements

Collisionless conditions have been assumed throughout. This allows the treatment of spin as a conserved quantity. Collisional phenomena are understood well in ICR and are separated into chemically reactive and nonreactive categories.[11] In ICR, an ensemble

of ions is observed after coherent excitation, and the signal is modeled in terms of an effective rotating electric monopole in the trap.[21] Nonreactive collisions lead to dephasing of the individual orbits, a decrease in the radius of the monopole, and a loss of signal; reactive collisions eliminate ions. Assuming that the collisions in the case proposed here are nonreactive, analysis of the reduced collision frequency for the Langevin potential (ion-induced dipole potential) gives room-temperature relaxation times of around 2 seconds for 100 amu ions of unity charge with pressure of 10^{-8} torr.[21] This corresponds to linewidths (and reduced collision frequencies) on the order of the frequency shifts to be measured. Therefore, vacuum better than 10^{-8} torr will be necessary.

Low temperature may help solve part of the collisional problem. In addition to slowing the motions of neutrals, low temperature may also induce neutrals to stick to the walls of the trap (cryopumping). The main advantage of working at low temperature, however, would be the decrease in Johnson noise in the preamplifier used for ICR detection. This factor combined with the large cyclotron radius will help make single ion detection practical.

4.3.e Coupling to rotation

Molecular rotation must still be incorporated into this theory. Previous work on the radiofrequency spectrum of H_2^+ polarized the sample optically, then probed the magnetic structure by observing the dissociation of the ion.[22] Since the cross section for photodissociation by a polarized beam depends on the magnetic state of the ion, excitation of the rf resonances between magnetic substates increases the ion decay rate. Experiments show that these magnetic substates include the coupling of electron spin to the two proton spins and to the rotation vector. An experiment to observe the magnetic resonance alone could work on cold H_2^+, in the limit of having only the lowest rotational level occupied. Alternatively, the coupling of the electron spin to the rotation could be observed by using para-H_2^+, for which the total nuclear spin quantum number I=0; Jefferts[23] discusses

photodissociation experiments of this type. Another possible direction to explore is that of detecting (purely) rotational transitions by the methods described here. For an ion with Zeeman splittings on the rotational spectrum there will be cyclotron shifts analogous to those described here for magnetic resonance. Since rotational spectroscopy of ions under collisionless conditions is impossible by straight microwave absorption techniques, such a method could augment the structural information available from magnetic resonance. Future work on the theory of detecting magnetic resonance via ICR must explore the coupling of spin to rotation, as well as to motional moments.

4.3.f Possible extensions

Since many "parent ions" of diamagnetic neutrals are paramagnetic, the techniques presented here could be applied to a large number of systems. In addition, this experiment need not be confined to the study of electron spins. Though nuclear spins have gyromagnetic ratios three orders of magnitude smaller than that of the electron, adjustment of the various parameters of the experiment could make the direct observation of nuclear resonances possible. The limits of low trapping voltages, large bottle fields, and high-resolution detection must be examined further to assess the possibility of NMR experiments.

The general scheme presented here is not the only way to separate spins via ICR. This work was originally motivated by the goal of creating spin-dependent cyclotron radius changes by the application of resonant gradients, enabling the detection of nuclear magnetic moments. In principle, any inhomogeneous field will couple spin to the spatial coordinates, making the spatial separation of spin states possible. The challenge is to find experiments, like the one presented here, which translate this possibility into easily observable quantities. This goal has been fulfilled, and the resulting proposals are described in the following chapter.

4.4 References

[1]Bowers, C.R., S.K. Buratto, P.J. Carson, H.M. Cho, J.Y. Hwang, L.J. Mueller, P.J. Pizarro, D.N. Shykind, and D.P. Weitekamp, SPIE Proc. **1435**, 36 (1991).

[2]Brown, L.S., and G. Gabrielse, Rev. Mod. Phys. **58**, 233 (1986).

[3]Van Dyck, R., Jr., P. Ekstrom, and H. Dehmelt, Nature **262**, 776 (1976).

[4]Van Dyck, R.S., Jr., P.B. Schwinberg, and H.G. Dehmelt, Phys. Rev. Lett. **38**, 310 (1977).

[5]Van Dyck, R.S., Jr., P.B. Schwinberg, and H.G. Dehmelt, in *New Frontiers in High Energy Physics*, edited by B. Kursunoglu, A. Perlmutter, and L. Scott (Plenum, New York, 1978).

[6]Van Dyck, R.S., Jr., P.B. Schwinberg, and H.G. Dehmelt, in Atomic Physics **9**, edited by R.S. Van Dyck, Jr., and E.N. Fortson (World Scientific, Singapore, 1984).

[7]Alford, J.M., P.E. Williams, D.J. Trevor, and R.E. Smalley, Int. J. Mass Spec. Ion Proc. **72**, 33 (1986).

[8]Abragam, A., Principles of Nuclear Magnetism (Oxford University Press, New York, 1989).

[9]Rabi, I.I., J.R. Zacharias, S. Millman, and P. Kusch, Phys. Rev. **53**, 318 (1938).

[10]Weltner, W., Jr., Magnetic Atoms and Molecules (Dover Publications, New York, 1989).

[11]Beauchamp, J.L., Ann. Rev. Phys. Chem. **22**, 527 (1971).

[12]Coe, J.V., J.C. Owrutsky, E.R. Keim, N.V. Agman, D.C. Hovde, and R.J. Saykally, J. Chem. Phys. **90**, 3893 (1989), and references therein.

[13]Weitekamp, D.P., and P.J. Pizarro, U.S. Patent No. 4,982,088 (1991).

[14]Gabrielse, G., Phys. Rev. A **27**, 2277 (1983).

[15]Gabrielse, G., and F.C. MacKintosh, Int. J. Mass Spectrom. Ion Processes **57**, 1 (1984).

[16]Schmidt, G., *Physics of High Temperature Plasmas*, 2nd ed. (Academic Press, New York), 1979.

[17]Poole, C.P., *Electron Spin Resonance: A Comprehensive Treatise on Experimental Techniques* (Wiley, New York), 1983.

[18]Cornell, E.A., R.M. Weisskoff, K.R. Boyce, R.W. Flanagan, Jr., G.P. Lafyatis, and D.E. Pritchard, Phys. Rev. Lett. **63**, 1674 (1989).

[19]Weisskoff, R.M., G.P. Lafyatis, K.R. Boyce, E.A. Cornell, R.W. Flanagan, Jr., and D.E. Pritchard, J. Appl. Phys. **63**, 4599 (1988).

[20]Press, W.H., B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical Recipes* (Cambridge University Press, Cambridge), 1986.

[21]Comisarow, M.B., Lect. Notes in Chem. **31**, 484 (1982).

[22]Richardson, C.B., K.B. Jefferts, and H.G. Dehmelt, Phys. Rev. **165**, 80 (1968).

[23]Jefferts, K.B., Phys. Rev. Lett. **20**, 39 (1968).

Chapter 5: Spin-Locked Internally Resonant Ion Cyclotron Excitation

The previous chapter suggests experiments which exploit single-ion sensitivity in the detection of the axial and cyclotron trapping frequencies. The magnitude of the frequency shifts produced, however, restricts the ready applicability of the proposals to the study of electron spin resonance. The ultimate goal of this dissertation is to develop experiments which can probe the magnetic resonance spectra of *nuclear* magnetic moments by using spin-dependent forces to modify trapped ion frequencies. This chapter presents the fulfillment of this goal and focuses on design issues for a promising method in which spin-dependent cyclotron acceleration is imposed. The resulting change in ion orbit is detected as a change in the axial trapping frequency.[1]

As in the electron g-factor measurement of Dehmelt[2,3], the shift in axial trapping frequency is proportional to the strength of a static magnetic bottle field gradient. However, that direct effect of a spin flip is impractically small for nuclear spin flips of ions. In the present case, the transverse magnetic moment is coupled to a radiofrequency gradient to provide an accelerating force. A precedent is M. Bloom's deflection of neutral molecular beams by radiofrequency field gradients (the "transverse Stern-Gerlach effect").[4,5] The effect desired is a continuous Stern-Gerlach force that will perform spin-dependent work on the ion, as opposed to the continuous Stern-Gerlach effect[6] produced by a magnetic bottle, which does no work on the ion.

Semiclassical and quantum mechanical derivations in this chapter present the conditions under which a magnetic field gradient modulated at both the Larmor and cyclotron frequencies will lead to cyclotron acceleration proportional to the transverse

magnetic moment of a coherent state of the particle and radiation field. This effect has been named *s*pin-*l*ocked *i*nternally *r*esonant *i*on *c*yclotron *e*xcitation (SLIRICE). In the presence of a magnetic bottle, the corresponding shift in the axial trapping frequency due to this spin-dependent work can be made much larger than the shift due directly to a spin flip.

This effect has been incorporated into a proposed experimental procedure in which the spin-flip probability, resulting from a period of high-resolution magnetic resonance, controls the presence or absence of a net axial frequency shift between two detection periods. The data reduction algorithm presented in Chapter 4 allows rapid conversion of the "before" and "after" signals from one or many trapped ions into a point of the magnetic resonance spectrum or interferogram. Simulated signals, including the anticipated noise from both the detection circuit and intrinsic quantum fluctuations in the number of spins flipped, indicate the method is practical.

5.1 Background: the transverse Stern-Gerlach experiment

The work presented later in this chapter shares a common root with the "transverse Stern-Gerlach" (TSG) experiment, first proposed by Bloom and Erdman in 1962.[4] The experiment, demonstrated on a beam of potassium atoms in 1967,[5] achieved spin-dependent acceleration of neutral atoms in a beam by applying a radiofrequency magnetic field gradient resonant with the Larmor frequency corresponding to the atoms' spin angular momentum. With a field of the form

$$\vec{B} = B_0\hat{z} + \left(G_x x\hat{x} + G_y y\hat{y}\right)\cos\omega t, \qquad [5.1]$$

the resonance condition $\omega = \omega_0 = -\gamma B_0$ causes the spin to be quantized transversely in the reference frame rotating about the **z** axis with angular velocity ω.[5] This is contrast to the conventional Stern-Gerlach experiment, where the quantization is along the **z** axis and the

spin-dependent force is due to gradients in B_z. The demonstration of the transverse Stern-Gerlach experiment was accomplished using a quadrupole field (produced by four anti-phase wires along the **z** axis, placed at the corners of a square in the transverse plane) with $G_x = G$ and $G_y = -G$.[5]

Bloom and Erdman proposed the extension of the transverse Stern-Gerlach experiment to ion beams using similar fields, and predicted (but did not derive concretely) the existence of $\Delta n = \pm 1$, $\Delta m_s = \pm 1$ transitions for $\omega = \omega_0 - \omega_c$ and $\Delta n = \pm 1$, $\Delta m_s = \mp 1$ transitions for $\omega = \omega_0 + \omega_c$; in the same paper, they announced the construction of such an apparatus to measure the magnetic moment of the free electron,[4] but the completion of this experiment was never reported. In this chapter it is demostrated that the rf field configuration suggested there does not have the desired effect of steady spin-dependent cyclotron acceleration, but that is achievable with modified rf fields. Eight years later, Enga and Bloom[7] proposed a TSG experiment for ion beams traveling down the axis of a helical quadrupole wire system. The quadrupole nature of the magnetic field gradient would provide the necessary spatial dependence, while, in the ion frame of reference, the helical twist of the four quadrupole wires would translate into the time dependence essential to achieve resonance. However, calculations done as part of this dissertation work failed to derive the expected linear, spin-dependent changes in cyclotron energy when a time-dependent quadrupole field alone was added to a Penning trap.

Numerical trajectory simulations were carried out to explore the feasibility of the TSG effect for trapped ions. The trajectories were carried out with a version of the program IRICE01.FOR (see Appendix A) modified to add the field due to a quadrupole wire arrangement parallel to the **z** axis of a Penning trap. The simulations integrated the six-dimensional system of equations of motion for the three spatial (Cartesian) coordinates (x,y,z) and the three semiclassical spin coordinates (μ_x,μ_y,μ_z). Note that the spin is scaled by a factor of 10^8 to magnify spin-dependent effects, enhancing their visibility in runs with limited computational time. The cyclotron quantum number for several 40 μs runs is

plotted for several simulation runs in Fig. 5.1; the corresponding transverse magnetic moments are depicted in Fig. 5.2 The solid line trace corresponds to excitation by a quadrupole field (of frequency $\omega = \omega_0 - \omega_c$) alone, and shows no cyclotron acceleration. The transverse magnetic moment in this case is modulated between $|\vec{\mu}|$ and zero, indicating that a quadrupole field does not lock the spin in the transverse plane, and hence cannot produce a TSG effect, contradicting the previous proposals.[4,7] The addition of a spatially homogeneous spin-locking field b cos $\omega_0 t$, however, corrects this deficiency. This field constrains the spin magnetic moment to have a non-zero transverse component at all times, locking the spin in the transverse plane. Viewed in the frame rotating about the **z** axis with angular frequency ω, the spin is locked in the **x** direction. The spin-down results with two different values of b show that doubling the spin-locking field from 0.005 T to 0.010 T increases the average transverse magnetization from $0.69|\vec{\mu}|$ to $0.80|\vec{\mu}|$. In turn, this increases the linear change in cyclotron quantum number (*i.e.*, the TSG effectiveness) by a factor of 2.5. The spin-up and spin-down runs with b = 0.010 T show equal but opposite changes in cyclotron quantum number, the desired spin-dependent effect.

Several, albeit limited, attempts have been made via numerical simulations, semiclassical calculations, and quantum mechanical calculations to improve the degree of spin locking with this configuration. When viewed in the rotating frame, loss of a spin lock arises even with an ideal locking configuration when the initial spin condition does not correspond to one of the spin eigenstates of the system. The rotating frame magnetization then precesses about the eigenvector axis. In the simulations presented above, the axial magnetization observed was symmetrical, indicating that the spin eigenvector in each case was in the transverse plane. However, runs with the spin initially in the **y** direction and at angles of $\pi/4$, $3\pi/4$, $5\pi/4$, and $7\pi/4$ relative to the **x** axis (in the transverse plane) failed to give perfect spin locks (no modulation of the transverse and axial magnetization magnitudes in the laboratory frame). The sections that follow produce

the desired effect with a different field configuration; both semiclassical and quantum mechanical derivations for this configuration are presented.

5.2 Semiclassical derivation of SLIRICE

Consider the case of a positive ion confined in a Penning trap. Choosing the large homogeneous magnetic field B_0 to define the positive **z** axis, and neglecting magnetron motion throughout this derivation, the ion's transverse motion is given by

$$\vec{x} = \rho_+\left[\cos(\omega_+ t + \phi_+)\hat{x} - \sin(\omega_+ t + \phi_+)\hat{y}\right] \qquad [5.2],$$

where the cyclotron radius ρ_+ and the cyclotron frequency ω_+ are both positive quantities, and ϕ_+ is the cyclotron phase relative to an ion on the positive **x** axis at time t = 0. Since the results being derived will be applied to nuclear magnetic resonance, spin will be expressed in terms of the gyromagnetic ratio, as is conventional in the NMR literature. Assuming that the ion has a spin-1/2 magnetic moment $\vec{\mu}$, of magnitude $\mu = |\gamma|\hbar/2$, which has been spin-locked in the **x**-**y** plane, the spin's motion is given semiclassically by

$$\vec{\mu} = \frac{|\gamma|\hbar}{2}\left[\cos(\omega_0 t + \phi_0)\hat{x} + \sin(\omega_0 t + \phi_0)\hat{y}\right], \qquad [5.3]$$

with no **z** component. This spin-lock is the transversely quantized spin eigenstate of the transverse Stern-Gerlach effect;[5] any field configuration which performs spatial work dependent on the transverse spin moment must preserve this spin lock. For a spin with a negative gyromagnetic ratio γ (*e.g.*, a proton), ω_0 is a positive quantity; the two (quantum mechanical) orientations of spin have phases that differ by π.

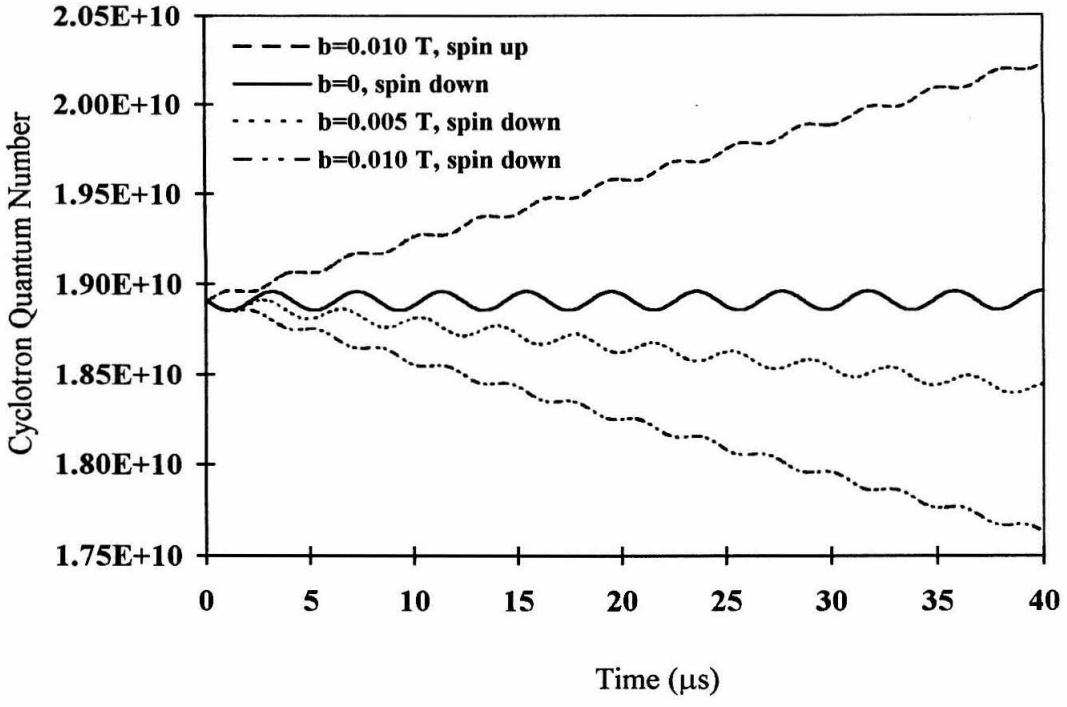


Fig. 5.1. Cyclotron quantum number changes due to TSG. Results from trajectory simulations with G = 0.023 T/m in a Penning trap with B_0 = 1 T, V_0 = 0.5 V, d = 1.1 cm. In each case, a 100 amu ion with a single positive charge and a proton magnetic moment is on the **x** axis at time t = 0, with cyclotron radius and axial amplitude both equal to 0.5 cm and negligible magnetron radius. The quadrupole field oscillates at the difference of the Larmor and cyclotron frequencies, while the spin-locking field of magnitude b oscillates at the Larmor frequency. The legend indicates the spin-locking field strength and the initial spin orientation for each run, where "up" and "down" are parallel and antiparallel, respectively, to the **x** axis. The simulation time step is 5×10^{-11} s; the spin magnetic moment is artificially scaled by a factor of 10^8 to make spin-dependent effects readily visible with limited computational time.

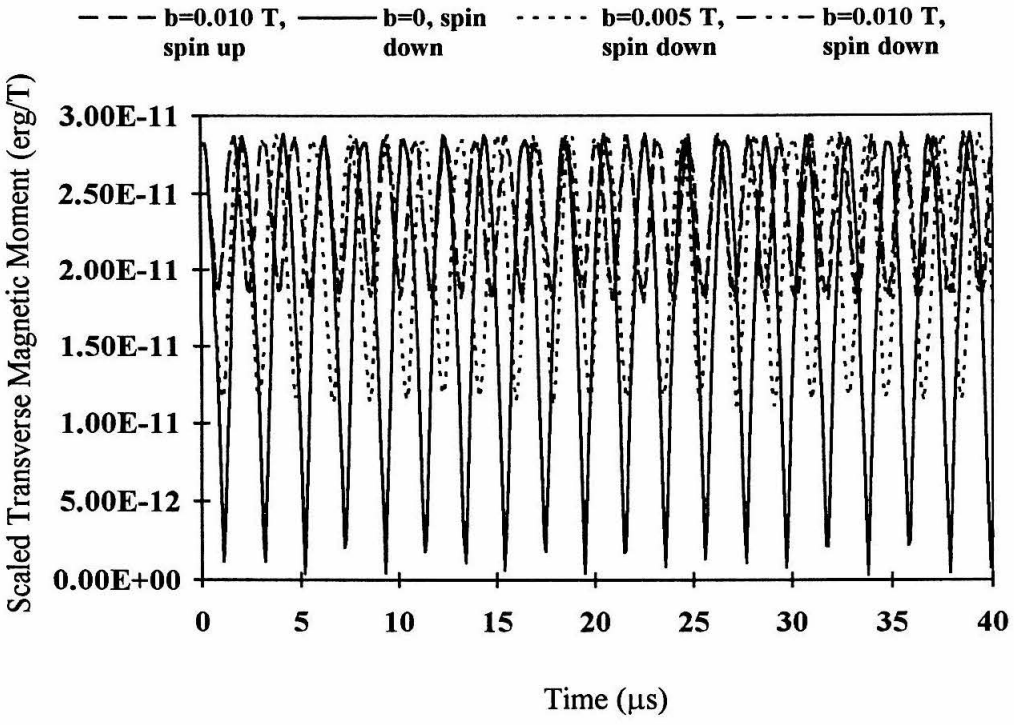


Fig. 5.2. Transverse magnetic moment during TSG. Results from trajectory simulations described in Fig. 5.1; the magnetic moment scaling by 10^8 mentioned in that caption sets the magnitude of the spin moment at 2.88×10^{-11} erg/T. The two b = 0.010 T curves overlap almost completely and are not distinguishable on this plot. The salient feature of this figure is the nearly total loss of transverse magnetic moment for the b = 0 curve.

The ultimate goal of this calculation will be to derive a force that accelerates or decelerates the cyclotron motion depending on the spin eigenstate's orientation. This force must be resonant with the cyclotron motion, and, by analogy to ion cyclotron resonance, tangential to the cyclotron orbit. That is, the force must be parallel (or antiparallel) to the direction of motion, so that the ion is continually pushed (or pulled). The force will thus be perpendicular to the position vector, and will be described by

$$\vec{F}_s \propto \pm\sin(\omega_+ t + \phi_+)\hat{x} \pm \cos(\omega_+ t + \phi_+)\hat{y}. \qquad [5.4]$$

5.2.a SLIRICE field configuration

Since the goal is the design of an ICR-like experiment with spin dependence, the resonant electric field of ICR is replaced by an oscillating magnetic gradient with components at the cyclotron and Larmor frequencies. This field is constructed by arranging two orthogonal quadrupole coils: one is parallel to the **x**-axis with current proportional to $\cos(\omega_0 t+\phi_1)\cos(\omega_+ t+\phi_2)$, and the other is directed along the **y**-axis with current proportional to $\cos(\omega_0 t+\phi_3)\cos(\omega_+ t+\phi_4)$., as shown in Fig. 3. The total magnetic field is now

$$\vec{B} = B_0\hat{z} + G(y\hat{y} - z\hat{z})\cos(\omega_0 t + \phi_1)\cos(\omega_+ t + \phi_2)$$

$$+G(z\hat{z} - x\hat{x})\cos(\omega_0 t + \phi_3)\cos(\omega_+ t + \phi_4) + b\cos\omega t. \qquad [5.5]$$

The greater number of adjustable parameters provides more design freedom than the single-frequency quadrupole originally suggested for TSG. A Helmholtz pair contributes the spin-locking field $b\cos\omega t\,\hat{x}$; this field has been accounted for already by the assumption of a transverse magnetic moment in Eq. [5.3], and will be neglected for the rest of this semiclassical calculation. The corresponding spin-dependent force on the ion is

$$\vec{F}_s = \vec{\nabla}(\vec{\mu}\bullet\vec{B}); \qquad [5.6]$$

this is approximated by

$$\vec{F}_s = (\vec{\mu}\bullet\vec{\nabla})\vec{B}. \qquad [5.7]$$

The expression for the force can be expanded and compared to the desired ICR-like form. The field's arbitrary phases can then be selected to make the spin-dependent force tangential to and resonant with the cyclotron motion. The resulting values for the phases are

$$\phi_1 = \phi_0 + \pi/2, \qquad [5.8a]$$

$$\phi_2 = \phi_+, \qquad [5.8b]$$

$$\phi_3 = \phi_0 + \pi, \qquad [5.8c]$$

and

$$\phi_4 = \phi_+ + \pi/2; \qquad [5.8d]$$

note that other (equivalent) solutions can be constructed by simply adding π to any two phases. With these values, the field reduces to

$$\vec{B} = B_0\hat{z} + b\cos\omega t + \frac{G}{2}\left\{(x\hat{x} - y\hat{y})\sin\left[(\omega_0 - \omega_+)t\right] - (x\hat{x} + y\hat{y} + 2z\hat{z})\sin\left[(\omega_0 + \omega_+)t\right]\right\}. \qquad [5.9]$$

The spin lock is lost at the nutation rate

$$\frac{\partial\mu_z}{\partial t} = -\frac{\gamma|\gamma|G\hbar\rho_+}{4}\sin\left[2(\omega_0 t + \phi_0)\right]\sin\left[2(\omega_+ t + \phi_+)\right]. \qquad [5.10]$$

Integration over time reveals the time dependence of the **z** component of the magnetization:

$$\mu_z(t) = -\frac{\gamma|\gamma|G\hbar\rho_+}{8}\times$$

$$\left\{\frac{1}{\omega_0 - \omega_+}\left\{\sin\left[2(\omega_0 - \omega_+)t + 2(\phi_0 - \phi_+)\right] - \sin\left[2(\phi_0 - \phi_+)\right]\right\}\right.$$

$$\left. + \frac{1}{\omega_0 + \omega_+}\left\{\sin\left[2(\phi_0 + \phi_+)\right] - \sin\left[2(\omega_0 + \omega_+)t + 2(\phi_0 + \phi_+)\right]\right\}\right\}. \qquad [5.11]$$

For a proton in a 1 T magnetic field with $\rho_+ = 0.5$ cm and G = 1 T/m, the amplitude of the terms in $\mu_z(t)$ is $6.29\times10^{-4}\,|\vec{\mu}|$, a negligible quantity, and the spin lock is conserved.

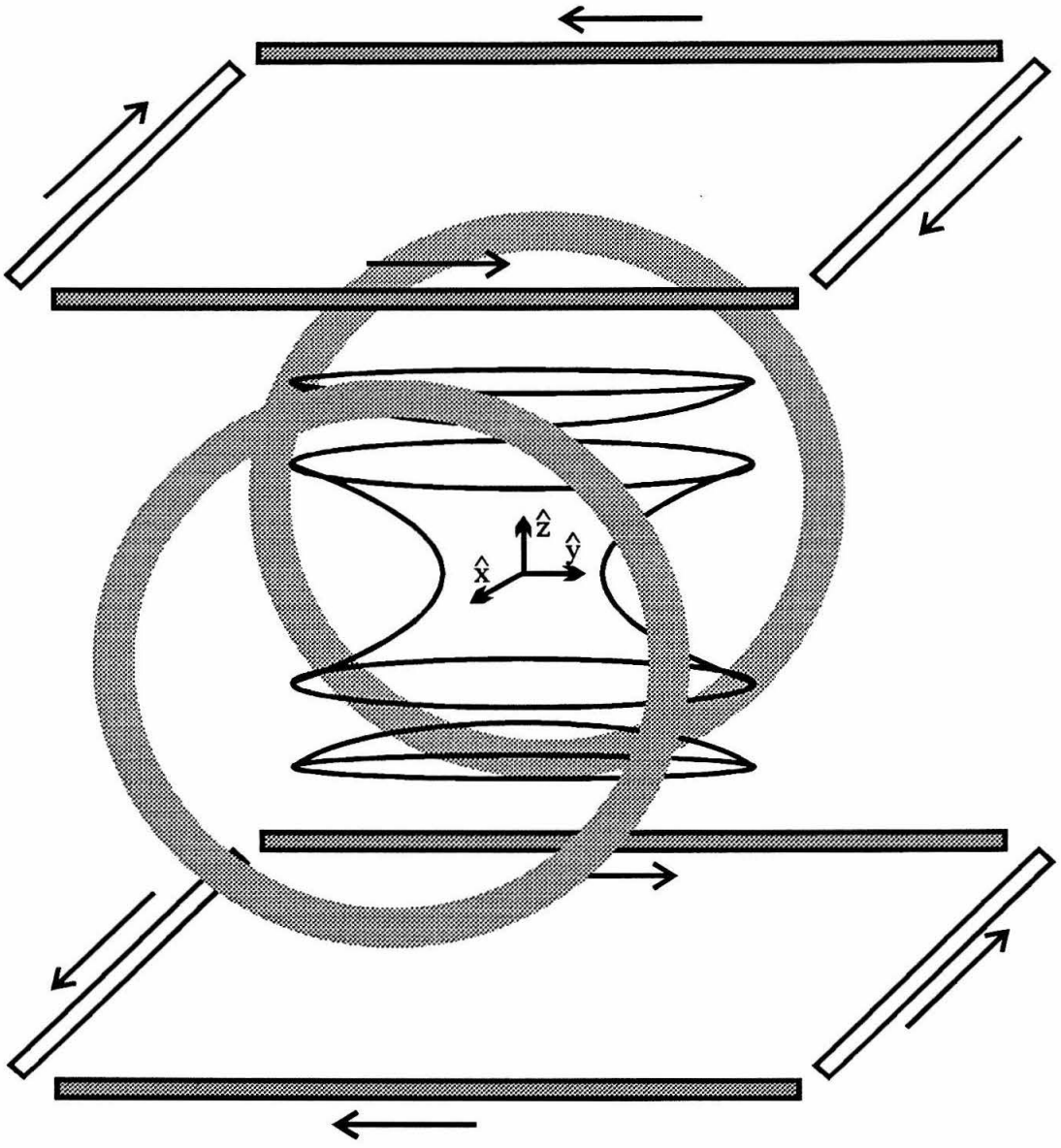


Fig. 5.3. Orthogonal quadrupole coil and Penning trap configuration for SLIRICE. The dark wires compose a quadrupole coil along the **y** axis, while the light wires are part of the coil parallel to the **x** axis. The arrows indicate the relative direction of the electrical current traveling through the wires. The vertical loops represent a Helmholtz pair producing the spin-locking field $b \cos\omega t\, \hat{x}$.

5.2.b Change in cyclotron radius and its monitoring via axial frequency shifts

The spin-dependent force, neglecting high-frequency (greater than ω) terms, is

$$\vec{F}_s = \mp\frac{|\gamma|\hbar G}{4}\left[\sin(\omega_+ t)\hat{x} + \cos(\omega_+ t)\hat{y}\right]. \qquad [5.12]$$

For excitation by this spin-dependent force small compared to the initial energy, the change in cyclotron energy due over a period τ is

$$\Delta E_+ = -\int_{\vec{x}(0)}^{\vec{x}(\tau)} \vec{F}\bullet d\vec{x} = \int_0^{\tau} \vec{F}\bullet\left(\frac{\partial\vec{x}}{\partial t}\right)dt = \mp\frac{|\gamma|\hbar G\omega_+\rho_+\tau}{4}\cos\phi_+. \qquad [5.13]$$

The corresponding change in cyclotron radius is

$$\Delta\rho_+ = \mp\frac{|\gamma|\hbar G\tau}{4m\omega_+}\cos\phi_+, \qquad [5.14]$$

the desired spin-dependent displacement, linear in time. The effect is depicted schematically in Fig. 5.4.

This shows that an ensemble of ions originally at radius ρ_+ with spins quantized along the **z** axis, which are accelerated to larger or smaller orbits depending on their transversely quantized spin state and their azimuthal position, can be monitored as a change in the axial frequency in the presence of a magnetic bottle. The axial frequency for each ion, including corrections to ω_z due to the magnetic bottle, can be written from Eq. [2.42] (using the NMR notation) as

$$\tilde{\omega}_z^{\pm} = \omega_z + \frac{B_2}{m\omega_z}\left(\mu_m \mp \frac{|\gamma|\hbar}{2}\right) + O\left(B_2^2\right) \qquad [5.15]$$

for spin quantum number $m_s = \pm 1/2$. The change in axial frequency due to a change in cyclotron radius is

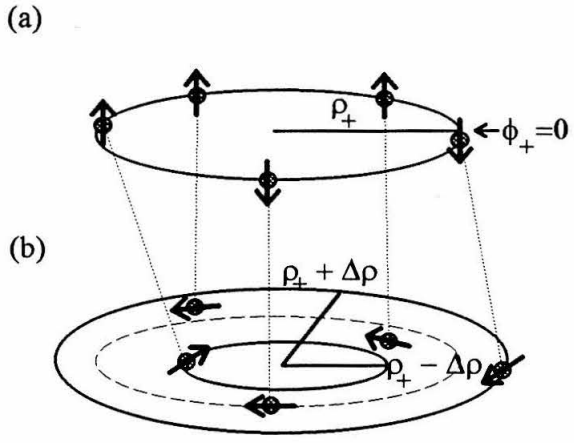


Fig. 5.4. Illustration of the IRICE effect. The circles represent ions; the arrows represent the orientation of each ion's spin eigenvector. (a) Prior to IRICE, the axis of spin quantization is the **z**-axis. Ions excited to the cyclotron radius ρ_+ by coherent ICR excitation have been dephased by B_2 during an initial axial detection period. (b) IRICE quantizes the spins transversely and accelerates the ions to new orbits. The effect scales with $\cos\phi_+$, as seen in Eq. 8., and is thus maximized with $\phi_+ = 0$ or π.

$$\delta = \tilde{\omega}_z(\rho_+ + \Delta\rho_+) - \tilde{\omega}_z(\rho_+). \qquad [5.16]$$

Neglecting both the direct spin term and the magnetron motion, and rewriting the mechanical magnetic moment in terms of the cyclotron radius, the axial frequency including the effect of the cyclotron radius in a magnetic bottle is

$$\tilde{\omega}_z^{\pm} = \omega_z + \frac{q\omega_+ B_2}{m\omega_z}\rho_+^2. \qquad [5.17]$$

The SLIRICE axial frequency shift, neglecting terms quadratic in $\Delta\rho_+$, is

$$\delta = \frac{q\omega_+ B_2 \rho_+}{m\omega_z}\Delta\rho_+. \qquad [5.18]$$

Inserting Eq. [5.14] for the cyclotron radius change yields the frequency shift

$$\delta = \mp \frac{qB_2|\gamma|\hbar G\rho_+ \cos\phi_+}{4m^2\omega_z}\tau. \qquad [5.19]$$

For a 100 amu ion with a proton magnetic moment in B_0 = 1 T, with B_2 = 1000 T/m², $\tilde{\omega}_z/2\pi$ = 26253.9 Hz, ρ_+ = 5 mm, ϕ_+ = 0, G = 200 G/cm, and τ = 1 sec, the spin-dependent change in the cyclotron radius is $\Delta\rho_+$ = 1.4 μm, with the corresponding axial frequency shift $\delta/2\pi$ = 0.4 Hz. This represents an improvement of five orders of magnitude when compared to the 4 μHz direct shift of the frequency caused by the spin in a magnetic bottle, as calculated from the full axial frequency above as well as from the perturbation expression for $\Delta\omega_z$, Eq. [3.26b].

The expression derived above is good as long as the components of the field gradient are exactly resonant with the Larmor and cyclotron frequencies. Resonance with the cyclotron frequency is achieved easily for a single mass as long as SLIRICE takes place in the absence of static field gradients which might broaden the cyclotron lines over an ensemble of ions. This, of course, requires that any experimental procedure which detects the SLIRICE radius change via axial frequency shifts in a magnetic bottle must keep the SLIRICE and axial detection periods separate in time, so that B_2 may be off in the former periods and on in the latter.

The assumed condition of spin resonance poses a problem, however, since a time-domain NMR experiment involves exciting all lines uniformly across some broad spectral region. Inability to observe off-resonance lines would require the use of continuous wave techniques, leading to a slow time-sequenced experiment in which the Larmor component of the radiofrequency field would be stepped in frequency from one shot to the next. Spin locking over a range of NMR line frequencies solves this problem allowing spin-dependent cyclotron acceleration to be incorporated into time-domain NMR experiments. It allows spins with Larmor frequencies across a broad spectrum to remain in phase with the spin-dependent force, allowing all lines to be excited nearly equally. A quantitative theory of

these offset effects is derived via a fully quantum mechanical calculation in the next section.

5.3 Quantum mechanical derivation of SLIRICE

The SLIRICE effect obtained semiclassically in the previous section will now be derived more systematically via a quantum mechanical rotating frame calculation similar to that used to evaluate ICR excitation in Chapter 3. Unlike the semiclassical derivation, this method will include the spin-locking field and resonance offsets explicitly. The rotating frame used is defined by the transformation $T = \exp(+iH_0t)$ with

$$H_0 = H_\rho + H_z - \omega I_z; \qquad [5.20]$$

the transverse and axial Hamiltonians are those defined in Chapter 3. The SLIRICE effect is produced by adding the rf field

$$\vec{B}_1 = G\left(y\hat{j} - z\hat{k}\right)\cos(\omega t + \phi_1)\cos(\omega_+ t + \phi_2)$$

$$+G\left(z\hat{k} - x\hat{i}\right)\cos(\omega t + \phi_3)\cos(\omega_+ t + \phi_4) + b\cos\omega t\ \hat{x}. \qquad [5.21]$$

The corresponding perturbative addition to the total Hamiltonian $H = H_0 + V$ is

$$V = \gamma\vec{I} \bullet \vec{B}_1 + \Delta\omega I_z, \qquad [5.22]$$

where $\vec{I}$ is the vector of spin operators (I_x, I_y, I_z) and the resonance offset is

$$\Delta\omega = \omega - \omega_0. \qquad [5.23]$$

Expansion of the perturbation yields

$$V = -\gamma G x I_x \cos(\omega t + \phi_3)\cos(\omega_+ t + \phi_4) + \gamma G y I_y \cos(\omega t + \phi_1)\cos(\omega_+ t + \phi_2)$$

$$+\gamma G z I_z\left[\cos(\omega t + \phi_3)\cos(\omega_+ t + \phi_4) - \cos(\omega t + \phi_1)\cos(\omega_+ t + \phi_2)\right]$$

$$+\gamma b I_x \cos\omega t + \Delta\omega I_z. \qquad [5.24]$$

5.3.a Rotating frame interaction

The interaction in the rotating frame is

$$\tilde{V} = e^{-iH_0 t/\hbar} V e^{iH_0 t/\hbar}. \qquad [5.25]$$

Expanding this yields

$$\tilde{V} = -\gamma G\tilde{x}\tilde{I}_x \cos(\omega t + \phi_3)\cos(\omega_+ t + \phi_4) + \gamma G\tilde{y}\tilde{I}_y \cos(\omega t + \phi_1)\cos(\omega_+ t + \phi_2)$$

$$+\gamma G\tilde{z}\tilde{I}_z\left[\cos(\omega t + \phi_3)\cos(\omega_+ t + \phi_4) - \cos(\omega t + \phi_1)\cos(\omega_+ t + \phi_2)\right]$$

$$+\gamma b\tilde{I}_x \cos\omega t + \Delta\omega\tilde{I}_z. \qquad [5.26]$$

The individual rotating frame operators are now tabulated:

$$\tilde{x} = e^{-iH_\rho t/\hbar} x e^{iH_\rho t/\hbar}$$

$$= \sqrt{\frac{\hbar}{2m(\omega_+ - \omega_-)}}\left[\left(a_+^\dagger + a_+\right)\sin\omega_+ t - \left(a_-^\dagger + a_-\right)\sin\omega_- t\right.$$

$$\left. + i\left(a_+^\dagger - a_+\right)\cos\omega_+ t + i\left(a_-^\dagger - a_-\right)\cos\omega_- t\right]; \qquad [5.27a]$$

$$\tilde{y} = e^{-iH_\rho t/\hbar} y e^{iH_\rho t/\hbar}$$

$$= \sqrt{\frac{\hbar}{2m(\omega_+ - \omega_-)}}\left[\left(a_+^\dagger + a_+\right)\cos\omega_+ t - \left(a_-^\dagger + a_-\right)\cos\omega_- t\right.$$

$$\left. - i\left(a_+^\dagger - a_+\right)\sin\omega_+ t - i\left(a_-^\dagger - a_-\right)\sin\omega_- t\right]; \qquad [5.27b]$$

$$\tilde{z} = e^{-iH_z t/\hbar} z e^{iH_z t/\hbar} = \sqrt{\frac{\hbar}{2m\omega_z}}\left[\left(a_z^\dagger + a_z\right)\cos\omega_z t - i\left(a_z^\dagger - a_z\right)\sin\omega_z t\right]; \qquad [5.27c]$$

$$\tilde{I}_x = e^{i\omega I_z t/\hbar} I_x e^{-i\omega I_z t/\hbar} = I_x \cos\omega t - I_y \sin\omega t; \qquad [5.27d]$$

$$\tilde{I}_y = e^{-i\omega I_z t/\hbar} I_y e^{i\omega I_z t/\hbar} = I_x \sin\omega t + I_y \cos\omega t. \qquad [5.27e]$$

These expressions can be inserted into $\tilde{V}$ to evaluate the general rotating frame interaction. The problem is made more specific by selecting phases (by inspection) that will lead to the desired spin-dependent cyclotron acceleration. The phases chosen here coincide with those determined semiclassically in the previous section (Eq. [5.8]), assuming that the arbitrary reference phases ϕ_0 (of the rf) and ϕ_+ (of the ion motion) are both set to zero. Keeping only those terms that are constant in the rotating frame, the average rotating frame interaction is

$$\tilde{V}^{(0)} \approx -\frac{\gamma G}{2}\sqrt{\frac{\hbar}{2m(\omega_+ - \omega_-)}}\left(a_+^\dagger + a_+\right)I_x + \frac{\gamma b}{2}I_x + \Delta\omega I_z. \qquad [5.28]$$

5.3.b Laboratory frame radius operator

Restating the expression for the rotating frame radius operator from Chapter 3,

$$\tilde{\rho} = e^{-iH_\rho t/\hbar}\rho e^{iH_\rho t/\hbar} = i\sqrt{\frac{2\hbar}{m(\omega_+ - \omega_-)}}\left(a_+^\dagger e^{-i\omega_+ t} - a_- e^{-i\omega_- t}\right). \qquad [5.29]$$

The radius operator is now transformed to the interaction frame, becoming

$$\underline{\underline{\rho}}(t) = e^{-i\tilde{V}^{(0)}t/\hbar}\tilde{\rho}e^{i\tilde{V}^{(0)}t/\hbar}$$

$$= \tilde{\rho} - \frac{\gamma G e^{-i\omega_+ t}}{2m(\omega_+ - \omega_-)}\left\{ I_x t - \frac{\Delta\omega}{\underline{\underline{\omega}}_{eff}^2}\left[\cos\left(\underline{\underline{\omega}}_{eff} t\right) - 1\right]I_y \right.$$

$$\left. + \frac{\Delta\omega}{\underline{\underline{\omega}}_{eff}^3}\left[\sin\left(\underline{\underline{\omega}}_{eff} t\right) - \underline{\underline{\omega}}_{eff} t\right]\left(\Delta\omega I_x - \underline{\underline{\omega}}_1 I_z\right)\right\}, \qquad [5.30]$$

where the operators

$$\underline{\underline{\omega}}_1 = -\frac{\gamma G}{2}\sqrt{\frac{\hbar}{2m(\omega_+ - \omega_-)}}\left(a_+^\dagger + a_+\right) + \frac{\gamma b}{2} \qquad [5.31]$$

and

$$\underline{\underline{\omega}}_{\text{eff}}^2 = \underline{\underline{\omega}}_1^2 + (\Delta\omega)^2 \qquad [5.32]$$

have been defined to make the equations more compact and to isolate important precessional frequencies, as will become apparent later in this calculation. In this expression, the use of $\underline{\underline{\omega}}_{\text{eff}}^2$ in the denominator is simply a notational convenience to simplify the result into trigonometric operator form; expansion of the trigonometric functions into their respective infinite series reveals that all operators appear with powers that are positive integers. The solution for the radius observed in the laboratory frame is

$$\rho(t) = \left\langle \Psi \middle| e^{-i\tilde{V}^{(0)}t/\hbar} e^{-iH_0 t/\hbar} \rho e^{iH_0 t/\hbar} e^{i\tilde{V}^{(0)}t/\hbar} \middle| \Psi \right\rangle = \left\langle \Psi \middle| \underline{\underline{\rho}}(t) \middle| \Psi \right\rangle. \qquad [5.33]$$

5.3.c Wavefunctions in the classical limit

The wavefunction $|\Psi\rangle$ is assumed to separate into the simple product of the cyclotron (n), magnetron (l), axial (k), and spin ($\pm$, for $m_s = \pm 1/2$) wavefunctions:

$$|\Psi\rangle = |\Psi_n\rangle \otimes |\Psi_l\rangle \otimes |\Psi_k\rangle \otimes |\Psi_\pm\rangle. \qquad [5.34]$$

Since the operators present in the current calculation do not have any axial dependence, the axial wavefunction will simply factor out. The nature of the spin wavefunction will be investigated in the following section. The limit of classically-sized (large) orbits will be assumed, and thus the cyclotron and magnetron parts of the wavefunction are those derived in Chapter 3, with the combined wavefunction of Eq. [3.36] subject to the phase relationships of Eqs. [3.41] and the amplitude constraints of Eqs. [3.42]. Given these conditions, the expectation value of the operator $\underline{\underline{\omega}}_1$ reduces to

$$\omega_1 = \langle \Psi | \underline{\underline{\omega}}_1 | \Psi \rangle$$

$$= -\frac{\gamma G}{2}\sqrt{\frac{\hbar}{2m(\omega_+ - \omega_-)}}\sum_n \left[c^*_{n+1}c_n\sqrt{n+1} + c^*_{n-1}c_n\sqrt{n}\right] + \frac{\gamma b}{2}$$

$$= -\frac{\gamma G}{2}\sqrt{\frac{\hbar}{2m(\omega_+ - \omega_-)}}\sum_n \left[(ic_n)^* c_n\sqrt{n+1} + (-ic_n)^* c_n\sqrt{n}\right] + \frac{\gamma b}{2}$$

$$= -\frac{\gamma G}{2}\sqrt{\frac{\hbar}{2m(\omega_+ - \omega_-)}}\sum_n \left[-i|c_n|^2\left(\sqrt{n+1} - \sqrt{n}\right)\right] + \frac{\gamma b}{2}. \qquad [5.35]$$

For large n, the sum collapses to zero and ω_1 reduces to the Rabi frequency

$$\omega_1 \approx \frac{\gamma b}{2}. \qquad [5.36]$$

A similar analysis shows that

$$\omega_1^2 = \langle \Psi | \underline{\underline{\omega}}_1^m | \Psi \rangle \approx \left(\frac{\gamma b}{2}\right)^m \qquad [5.37]$$

and

$$\omega_{eff}^2 = \langle \Psi | \underline{\underline{\omega}}_{eff}^2 | \Psi \rangle \approx \left(\frac{\gamma b}{2}\right)^2 + (\Delta\omega)^2. \qquad [5.38]$$

These large n results reduce the observed laboratory frame radius to

$$\rho(t) = \rho_+ e^{-i\omega_+ t} + \rho_- e^{-i\omega_- t}$$

$$-\frac{\gamma G e^{-i\omega_+ t}}{2m(\omega_+ - \omega_-)}\left\{\langle \Psi_\pm | I_x | \Psi_\pm \rangle t - \frac{\Delta\omega}{\omega_{eff}^2}\left[\cos(\omega_{eff} t) - 1\right]\langle \Psi_\pm | I_y | \Psi_\pm \rangle .\right.$$

$$\left. + \frac{\Delta\omega}{\omega_{eff}^3}\left[\sin(\omega_{eff} t) - \omega_{eff} t\right]\left(\Delta\omega \langle \Psi_\pm | I_x | \Psi_\pm \rangle - \omega_1 \langle \Psi_\pm | I_z | \Psi_\pm \rangle\right)\right\}. \qquad [5.39]$$

5.3.d Spin eigenstates of the SLIRICE system

This calculation has as its goal deriving the conditions under which spin-1/2 wavefunctions may be detected via linear changes in the cyclotron radius. Experimentally, the SLIRICE procedure will measure the spin (as long as the cyclotron radius changes are larger than the ion deBroglie wavelength, a very small quantity for atomic ions) in the sense that it will project an arbitrary initial spin state into one of the SLIRICE eigenstates. Therefore, a calculation of this effect will succeed only if the spin eigenstates of the system are evaluated correctly. To this end, consider the spin eigenstates $|\Psi_{\pm}\rangle$ for a system quantized along the **u** axis (*i.e.*, the eigenstates of I_u), with

$$\hat{u} = \hat{z}\cos\theta + \hat{x}\sin\theta\cos\phi + \hat{y}\sin\theta\sin\phi. \qquad [5.40]$$

These are

$$|\Psi_+\rangle = \cos\frac{\theta}{2}e^{-i\phi/2}|+\rangle + \sin\frac{\theta}{2}e^{i\phi/2}|-\rangle; \qquad [5.41a]$$

$$|\Psi_-\rangle = -\sin\frac{\theta}{2}e^{-i\phi/2}|+\rangle + \cos\frac{\theta}{2}e^{i\phi/2}|-\rangle. \qquad [5.41b]$$

The angles θ and ϕ are defined in Fig. 5.5 and $|\pm\rangle$ are the eigenstates of I_z. Reevaluating the expression for the observed radius with these spin wavefunctions,

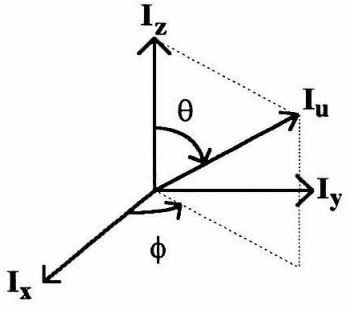


Fig. 5.5. Coordinate system for spin quantization along the **u** axis.

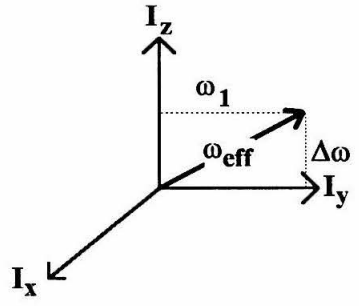


Fig. 5.6. Coordinate system for spin quantization in SLIRICE.

$$\rho^{(\pm)}(t) = \rho_+ e^{-i\omega_+ t} + \rho_- e^{-i\omega_- t}$$

$$\mp \frac{\gamma G\hbar e^{-i\omega_+ t}}{4m(\omega_+ - \omega_-)} \left\{ t\sin\theta\cos\phi - \frac{\Delta\omega}{\omega_{eff}^2}\left[\cos(\omega_{eff} t) - 1\right]\sin\theta\sin\phi \right.$$

$$\left. + \frac{\Delta\omega}{\omega_{eff}^3}\left[\sin(\omega_{eff} t) - \omega_{eff} t\right](\Delta\omega\sin\theta\cos\phi - \omega_1\cos\theta) \right\}. \qquad [5.42]$$

Since the transverse spin dependence of the rotating frame interaction $\tilde{V}^{(0)}$ consisted solely of I_x terms, it is expected that the SLIRICE eigenstates will lie along

$$\phi = 0, \qquad [5.43]$$

collapsing the radius expression to

$$\rho^{(\pm)}(t) = \rho_+ e^{-i\omega_+ t} + \rho_- e^{-i\omega_- t} \mp \frac{\gamma G\hbar e^{-i\omega_+ t}}{4m(\omega_+ - \omega_-)} \left\{ t\sin\theta\cos\phi \right.$$

$$\left. + \frac{\Delta\omega}{\omega_{eff}^3}\left[\sin(\omega_{eff} t) - \omega_{eff} t\right](\Delta\omega\sin\theta - \omega_1\cos\theta) \right\}. \qquad [5.44]$$

The goal of cyclotron radius changes that are exclusively linear in time will be realized only if

$$\frac{\Delta\omega}{\omega_{eff}^{3}}\left[\sin(\omega_{eff}t)-\omega_{eff}t\right](\Delta\omega\sin\theta-\omega_1\cos\theta)=0; \qquad [5.45]$$

the unique solution to this equation is

$$\theta=\tan^{-1}\left(\frac{\omega_1}{\Delta\omega}\right). \qquad [5.46]$$

Having evaluated the spin wavefunction axis of quantization (depicted in Fig. 5.6), the radius observed is readily calculated to be

$$\rho^{(\pm)}(t)=\rho_+e^{-i\omega_+t}+\rho_-e^{-i\omega_-t}\mp\frac{\gamma G\hbar e^{-i\omega_+t}}{4m(\omega_+-\omega_-)}\frac{\omega_1}{\omega_{eff}}t. \qquad [5.47]$$

The term depending linearly on time is of the same frequency and phase as the initial cyclotron radius and thus represents a change of ρ_+ exclusively, leaving the magnetron radius intact. The cyclotron radius is changed by

$$\Delta\rho_+^{(\pm)}(t)=\mp\frac{\gamma G\hbar}{4m(\omega_+-\omega_-)}\frac{\omega_1}{\omega_{eff}}t \qquad [5.48]$$

for spin in the $|\Psi_\pm\rangle$ eigenstates.

Since $\omega_+ \gg \omega_-$, this expression is equivalent to the semiclassically derived radius change of Eq. [5.14]. The two different approaches yield the same magnitude for the radius change, but provide additional different, and complementary, pieces of information. Whereas the semiclassical simulation explains the cyclotron phase dependence of the effect, the quantum mechanical derivation elucidates off-Larmor-resonance behavior. This behavior allows the application of SLIRICE to Fourier transform NMR spectroscopy over sufficiently large Larmor frequency bandwidths to study interesting problems. For example, with a spin-locking field strength b = 0.005 T (50 G) and a proton magnetic moment, the Rabi frequency is $\omega_1/2\pi$ = 106 kHz, and the SLIRICE efficiency ω_1/ω_{eff} is greater than 90% over a 100 kHz bandwidth. This would be adequate to cover essentially all proton resonances in any diamagnetic ion and all resonances within a particular electron spin level of a paramagnetic ion.

5.4 Semiclassical numerical trajectory simulations

The SLIRICE effect has been studied extensively through numerical trajectory simulations with the FORTRAN program IRICE01.FOR (see Appendix A). As was the case with the TSG simulations earlier in this chapter, the spatial and semiclassical spin equations of motion were integrated, and the spin was scaled by a factor of 10^8 to magnify spin-dependent effects and bring their observation into the reach of program runs with limited computational time. The motion of a 100 amu, singly charged, positive ion with a proton magnetic moment was simulated in a Penning trap with B_0 = 1 T, V_0 = 0.5 V, d = 1.1 cm, G = 0.01 T/cm, and b = 0.005 T over the course of 40 μs. The Larmor frequency ($\omega_0/2\pi$) is 42.58 MHz, and the Rabi frequency ($\omega_1/2\pi$) is 106 kHz. Both the axial amplitude and cyclotron radius were 0.5 cm, while the magnetron motion was zero.

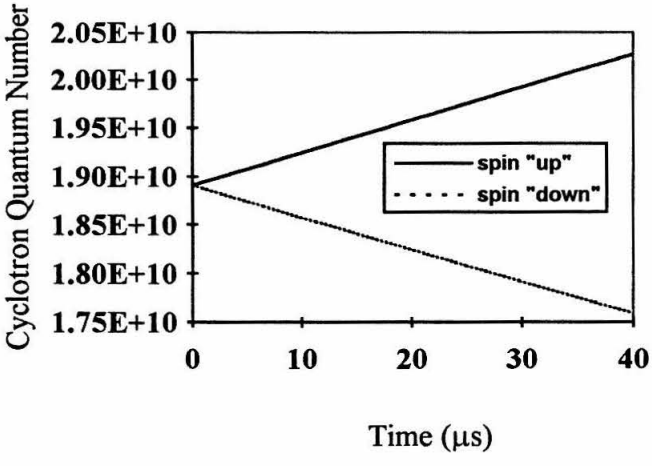


Fig. 5.7. SLIRICE cyclotron quantum number change on resonance, as described in the text.

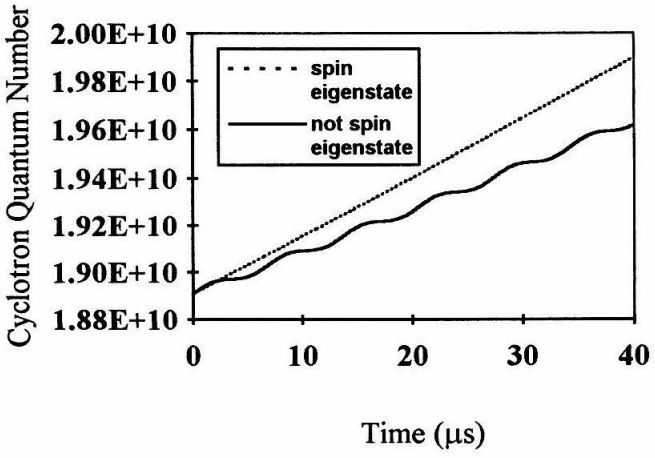


Fig. 5.8. SLIRICE cyclotron quantum number change off resonance, as described in the text.

Fig. 5.7 shows the cyclotron quantum number during an on-resonance ($\Delta\omega = 0$) SLIRICE period for ions in the "up" and "down" eigenstates of spin ($\pm I_x$ in the rotating frame). Spin-dependent cyclotron acceleration creates the expected linear changes in quantum number. (The integration time step in these runs is 5×10^{-11} s.) The transverse magnetization remains constant, indicating perfect spin locks.

Figs. 5.8 and 5.9 illustrate off-resonance behavior with $\Delta\omega/2\pi = 100$ kHz. (The integration time step in these simulations is 5×10^{-12} s.) The ion whose traces are marked "not spin eigenstate" starts the simulation with its spin along I_x. Since this is not the spin eigenstate (see Fig. 5.6 for the axis of quantization), the spin precesses about the eigenstate direction and is thus not locked, as observed in the deep modulation of the transverse magnetization (Fig. 5.9). The cyclotron radius change is not linear, but is modulated at the same frequency as the transverse magnetization. The ion marked "spin eigenstate" begins the simulation with its spin along the axis of quantization. It remains spin-locked (its transverse magnetization is constant), and its cyclotron change is linear in

time and 39% larger than that of the ion whose spin is not locked. As described in the nex section, SLIRICE will be used for fixed times and these small inefficiencies due to off-resonance behavior have no substantial effect on the ability to classify trajectories as needed to extract the NMR signal.

5.5 Proposed experimental procedure

5.5.a Time line for the proposed experiment

SLIRICE may be incorporated as the central feature of a proposed experimental magnetic resonance procedure by inserting it between two periods of axial frequency detection. Cyclotron radius changes due to IRICE are then monitored as axial frequency differences between the two detection periods, each carried out in the presence of a magnetic bottle, which converts the radius changes into frequency changes. The NMR information is encoded by introducing a period of spin evolution within the IRICE period.

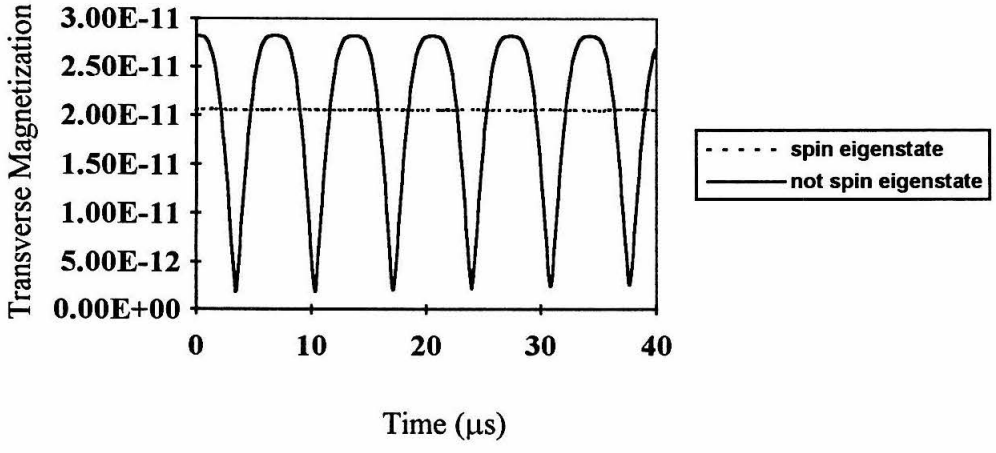


Fig. 5.9 Transverse magnetization during off-resonance SLIRICE, as described in the text.

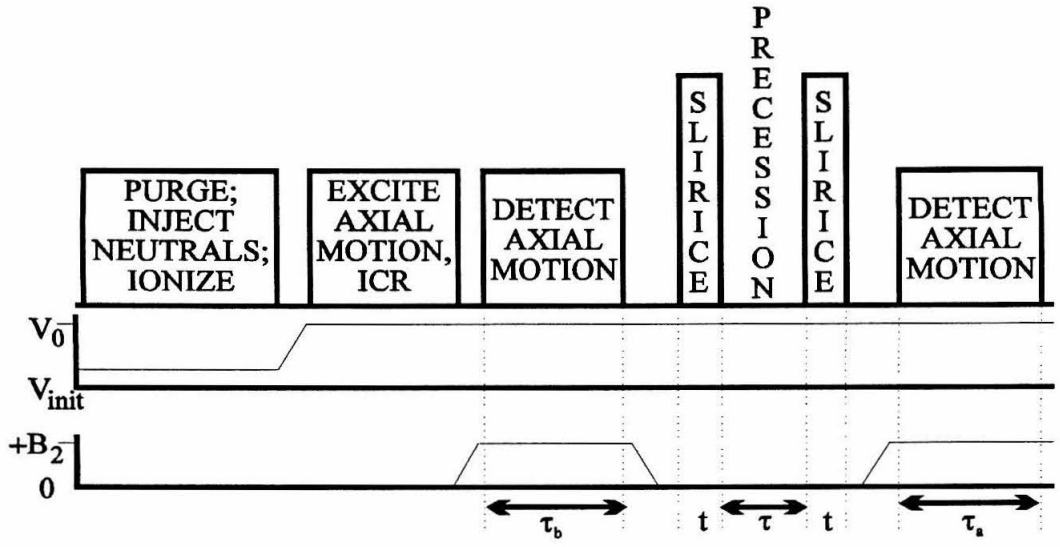


Fig. 5.10. Time line for proposed experiment incorporating the SLIRICE effect.

Inhomogeneity of the NMR spectrum is avoided by switching off the magnetic bottle during the spin evolution and the IRICE period. The spin evolution results in a certain probability of flipping the transverse orientation of the spin; this probability can then be modulated by varying the timing or frequency conditions of an NMR experiment. For example, IRICE could be split into two periods separated by spin evolution in B_0 during the variable time t_1. Fourier transformation of the resulting interferogram would yield the NMR spectrum. Alternately, a simpler (though slower) implementation could involve the sweeping of the Larmor frequency term in the IRICE field, yielding a continuous wave experiment. NMR lines would then be detected as peaks in the magnitude of the axial frequency shifts detected.

Fig. 5.10 shows the proposed experimental sequence. After purging the trap and injecting neutrals, ions are created in a low trapping voltage V_{init} which is swept adiabatically to the final value V_0. This procedure helps constrain the range of ion axial amplitudes present. The approach followed to overcome the inhomogeneous distribution of the axial frequency and to extract the axial frequency shifts involves convolving spectra

taken before and after the SLIRICE/NMR period, in a manner similar to the signal processing method proposed for direct monitoring of ESR in Chapter 4 (Fig. 4.2). The multiplication of the signal from the "before" axial detection period by the signal from the "after" period, with the "after" time axis reversed, produces a signal whose Fourier transform includes a peak at the sum of the "before" and "after" frequencies. Since the magnetic bottle is always positive in this proposal, the time-reversed product algorithm is adjusted to yield the difference of the frequencies, rather than the sum used in the ESR case. The derivation of the signal processing algorithm in Section 4.1.b still applies, but the time-reversed product is now

$$S_n^{ba}(\tau, t_1) = \tilde{S}_n^b(\tau_b)\,\tilde{S}_n^a(\tau_{max} - \tau_a). \qquad [5.49]$$

This differs from the Chapter 4 expression in the lack of complex conjugation of the time-reversed "after" signal.

The two SLIRICE periods are assumed to be in phase with each other, so that the precession period between them leads to a definite probability of a spin "flip." A "flip" occurs when the second SLIRICE period interrogates the spin state and the spin is not projected back into the state it was originally in during the first SLIRICE period. The spin evolution during the free precession period governs the probability P of this occurrence, which can be calculated as

$$P = \frac{1}{2}\left(\frac{\omega_1}{\omega_{eff}}\right)^2 \left[1 - \cos(\Delta\omega t_1)\right]. \qquad [5.50]$$

When the spin is flipped, the net SLIRICE cyclotron radius change and, hence, the net axial frequency shift detected is zero. Cases where the spin is not flipped lead to twice the single SLIRICE period shift δ. Since this shift is modulated by each ion's cyclotron phase, and since no cyclotron coherence is assumed in this proposal, the time-reversed products will consist of a sharp center peak at zero frequency and a continuous distribution of sidebands bounded by $\pm 2\delta$.

5.5.b Simulations

The experiment proposed above has been simulated with the FORTRAN program AXIALIR8.FOR, included in Appendix B. Singly charged, positive, 100 amu ions at room temperature (298 K) were created as if by ionization with an electron beam of 100 μm radius traveling down the **z** axis of a Penning trap with B_0 = 1 T, d = 1.1 cm, and V_{init} = 0.05 V; the trapping voltage was subsequently raised to 0.5 V. On the average, 83 ions were created in each shot, of which 55 were trapped and 12 had axial frequencies within the 205 Hz signal bandwidth (enforced by a filter to avoid oversampling). The signal and noise were modelled to correspond to a 100 MΩ impedance maintained at 2 K and mixed down into the Nyquist bandwidth with a 26000 Hz reference frequency. Each signal consisted of 2048 points sampled every 4.88 ms for each trace in t_1. Signal averaging was accomplished by adding 200 shots for every t_1 point. There were 32 points in t_1, sampled in 12.5 μs intervals, to give an NMR resolution of 2.5 kHz and a bandwidth of 80 kHz. A radiofrequency gradient field with G = 2 T/m was applied over a 1 s interval in each SLIRICE period, while the strength of the spin-locking field was b = 0.05 T. Adiabatic sweep times of 0.1 s were employed for both the trapping voltage and the magnetic bottle, and the bottle strength B_2 peaked at 1000 T/m^2. The simulation set the Larmor resonance offset of half the ions at 15 kHz, and that of the other half at 30 kHz, thereby mimicking a two line spectrum.

Fig. 5.11 shows the time-reversed products (after Fourier transformation) $S(\omega,t_1)$, for whose calculation a noise blanking threshold of three standard deviations above the mean was employed. The center peaks are narrow and readily visible, as expected, whereas the sideband signal intensity is diffused due to the full range of cyclotron phases present. The NMR interferogram $S(t_1)$ was constructed by first applying a baseline correction to the time-reversed products: the minimum value in each t_1 trace was subtracted from the entire trace. The integral of each trace was then calculated, and the traces were divided by their respective integrals in order to normalize the traces and remove possible effects from statistical fluctuations in the number of ions. The sideband regions were then integrated (from -0.9 Hz to -0.2 Hz for the left sideband, and 0.2 Hz to 0.9 Hz for the right sideband), their integrals were added together, and the integral of the central peak was subtracted from this sum to yield the NMR interferogram $S(t_1)$, shown in Fig. 5.12. This was Fourier transformed to reveal the NMR spectrum $S(\omega)$ included in Fig. 5.13. Note that this resolves the simulated lines at 15 kHz and 30 kHz well, and with an ample signal-to-noise ratio.

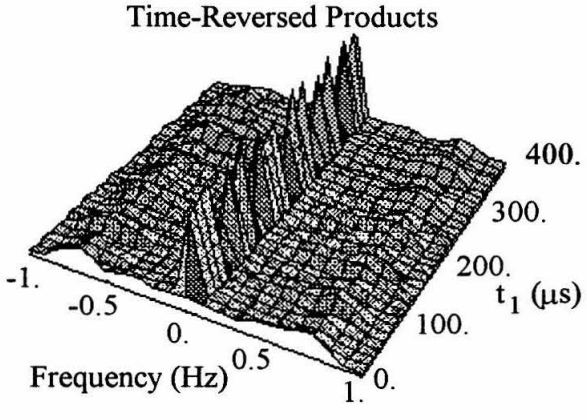


Fig 5.11. Time-reversed products $S(\omega,t_1)$ calculated in the simulation described in the text.

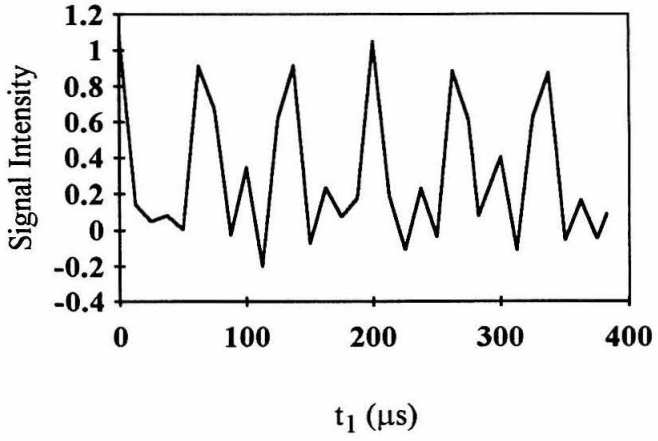


Fig. 5.12 NMR interferogram $S(t_1)$ for the simulation described in the text.

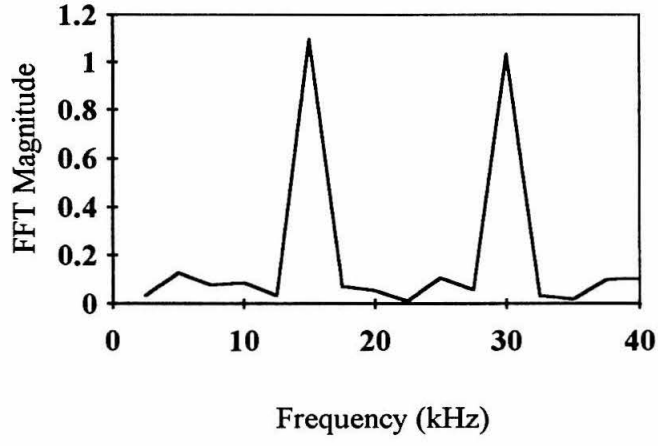


Fig. 5.13 NMR spectrum $S(\omega)$ for the simulation described in the text. The simulated lines at 15 kHz and 30 kHz are resolved. A peak at zero frequency, due to residual DC offsets, has been removed from the plot.

5.6 Discussion

A novel method for performing NMR spectroscopy on trapped ions has been designed. In contrast to the methods proposed for ESR in Chapter 4, where (1) the direct axial frequency shift due to the spin interaction with a magnetic bottle and (2) spin-dependent work on ion motion performed by a train of axially synchronized spin flips were used to encode the ESR spectrum into trapped ion motional frequencies, SLIRICE is a *continuous* Stern-Gerlach experiment leading to the separation of spin packets via changes in the cyclotron radius.

The technique proposed extends the applicability of NMR to gas phase trapped ions by circumventing the insensitive method of detection through Faraday law methods, substituting the single ion sensitivity of trapped ion detection. When applied to the elucidation of NMR spectra, this single-ion-sensitivity experiment stands in sharp contrast to conventional Faraday-law detection methods for NMR, which have a detection limit of $\approx 10^{18}$ spins. SLIRICE thus promises a sensitivity improvement of up to 18 orders of magnitude in the detection of NMR spectra. Since the observation time can be seconds, the potential frequency resolution is many orders of magnitude better than that of ion beam spectroscopy, which is transit-time limited.

5.7 References

[1]Pizarro, P.J., and D.P. Weitekamp, Bull. Mag. Reson. **14**, 220 (1992).

[2]Van Dyck, R.S., Jr., P.B. Schwinberg, and H.G. Dehmelt, in *New Frontiers in High Energy Physics*, edited by B. Kursunoglu, A. Perlmutter, and L. Scott (Plenum, New York, 1978).

[3]Van Dyck, R.S., Jr., P.B. Schwinberg, and H.G. Dehmelt, in Atomic Physics **9**, edited by R.S. Van Dyck, Jr., and E.N. Fortson (World Scientific, Singapore, 1984).

[4]Bloom, M., and K. Erdman, Can. J. Phys. **40**, 179 (1962).

[5]Bloom, M., E. Enga, and H. Lew, Can. J. Phys. **45**, 1481 (1967).

[6]Dehmelt, H., Z. Phys. D **10**, 127 (1988).

[7]Enga, E., and M. Bloom, Can. J. Phys. **48**, 2466 (1970).

Appendix A: Trajectory Simulation Programs

The complexity of the equations of motion of an ion a Penning trap subjected to various electric and magnetic fields with time and space dependence does not allow a classical analytical solution of most interesting problems. In such cases, the equations can be solved numerically via an iterative integration of individual ion trajectories for given sets of initial conditions. A good simulation program can be useful in determining the stability of trap configurations, extracting the frequencies of the various ion modes, and understanding the requirements and results of hypothesized experiments.

The first program presented here is TRAJ.FOR, written in FORTRAN to run on DEC VAX/VMS systems. The central integration algorithm is a three dimensional, fourth order, five point Runge-Kutta scheme as modified by Zonneveld.[1] Its coefficients take into account the full coupling of positions, velocities, and accelerations present in the second order differential equations of motion. TRAJ.FOR was easily modified to calculate ion trajectories in any combination of magnetic and electric fields; these fields may depend on space and time in any arbitrary manner. The version included here simulates the application of a magnetic bottle field gradient. The program reads its input values from a parameter file containing such quantities as initial time, position, and velocity, program step size, coefficients of common electric and magnetic field terms, and trap characteristics. Output includes the final time, position, and velocity, as well as two data files. One of these includes a user specified number of points at the beginning and end of the simulation, akin to "before" and "after" snapshots. The other output file is meant to be Fourier transformed: it contains a user-specified number of points from throughout the entire simulation, spaced by a sampling time determined from the user-

specified frequency resolution of the spectrum. The inverse of this resolution is the length of time the program simulates and the bandwidth is the resolution times one half the number of points sampled. A separate program, TRAJ_FT.FOR, processes the two output files; this is the second program presented in this Appendix. This program is kept separate from the more time consuming integration program; the simulation is run as a batch job, while the plotting and processing are handled interactively. TRAJ_FT.FOR is driven by interactive responses from the user to plot and Fourier transform the data. It can plot the "before" and "after" snapshots of the **x**, **y**, and **z** motions versus time, as well as a convenient **x-y** plot (in the plane of the cyclotron and magnetron motions). It can also apply a Fast Fourier Transform (FFT) algorithm[2] to transform the **x**, **y**, and **z** motions, plot these transforms, and find peaks. All graphics are handled by FORTRAN calls to the VMS implementation of the international standard Graphical Kernel System (GKS): the program was therefore easily transported to other systems.

The third program included in this Appendix is IRICE01.FOR, which is based on TRAJ.FOR. This trajectory simulation extends the integration space to include three spin coordinates in addition to the three spatial coordinates. In this fashion, a semiclassical picture of quantum mechanical effects can be achieved by introducing spin degrees of freedom into the problem of an ion stored in a Penning trap.

The simulation reproduces results from a broad variety of experiments. The step size required depends on the complexity of the fields used in a given experiment. Motion in a homogeneous magnetic field only requires as few as 50 to 100 steps per cyclotron orbit, while motion in a Penning trap and strong magnetic bottle (e.g., 1000 G/cm^2) may require as many as 1000 steps per cyclotron orbit for full convergence (e.g., to 1 Hz accuracy). Use of an adaptive step size algorithm has been avoided to minimize computational overhead time; since a typical project involves several simulations with equivalent fields, a few initial test runs suffice to determine an adequate step size.

A.1 References

[1]Zonneveld, J.A., *Automatic Numerical Integration* (Mathematical Centre Tracts, vol. 8, Mathematisch Centrum, Amsterdam), 1964.

[2]Press, W.H., B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical Recipes* (Cambridge University Press, Cambridge), 1986.

A.2 TRAJ.FOR, written for VAX/VMS systems

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C   TRAJ - Pedro J. Pizarro, 1/26/89                                   C
C                                                                      C
C       Given a desired frequency resolution RES, a number of          C
C       Fourier spectrum points NPTSFT, and a trajectory               C
C       simulation time step DT, along with initial values             C
C       for time, positions and velocities (all in the                 C
C       file TRAJ.PAR):      TRAJ  computes classical ion              C
C       trajectories and saves NPTSFT points spaced in time            C
C       so that the Nyquist frequency of their FFT will be             C
C       RES*NPTSFT/2.  Saves these points in FILE2, with               C
C       NPTSFT and FNYQST=RES*NPTSFT/2 in the first line.              C
C       Saves NARRAY-long records of the beginning and end             C
C       of the trajectory in FILE1, with NARRAY in                     C
C       the first line of the file, and tmin1, tmax1,tmin2,            C
C       and tmax2 in the last line.  Both data files are               C
C       unformatted, to be processed and plotted by traj_ft.           C
C                                                                      C
C   PJP - revisions - 4/17/89                                          C
C                                                                      C
C       Added capability of ICR in X or Y direction, by CW             C
C       or pulse.                                                      C
C                                                                      C
C   PJP - revisions - 7/6/89                                           C
C                                                                      C
C       Added common blocks /ion/ and /rkz4/.                          C
C                                                                      C
C   PJP - revisions - 7/12/89                                          C
C                                                                      C
C       Program now writes new versions of TRAJ.PAR and                C
C       TRAJB2.PAR at the end of the trajectory simulation.            C
C                                                                      C
C   PJP - revisions - 8/24/89                                          C
C                                                                      C
C       No longer writes new version of TRAJ.PAR (commented            C
C       out).  Only t,x,y,z,vx,vy,vz written to TRAJB2.PAR.            C
C                                                                      C
C   PJP - revisions - 10/9/89                                          C
C                                                                      C
C       Now calculates quantum numbers n,k,l for the initial           C
C       and final x,v.  Lines relevant to this change are set          C
C       aside between pairs of C####.                                  C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

```
      program traj

      parameter (nmax=3)
      parameter (nvar=3)
      double precision t,t1,dt,res,fnyqst,v(nmax),dv(nmax),
     1   x(nmax),q,m,magmom,spin,v0,d,b0,gradb,grade,b2,
     2   cwfreq,cwx,cwy,pulsex,pulsey,pi
C####
     3   ,hbar,wz,wc,wm,wp,quantn,quantk,quantl
C####
      real*4 tmin1,tmax1,tmin2,tmax2
      integer ncount,narray,nptsft,nperft,nft,nstep,nnext
      character*60 file1,file2

C
C     Common blocks (added on 7/6/89)
C
      common/ion/ q,m,magmom,spin,v0,d,b0,b2,gradb,grade,cwfreq,
     1   cwx,cwy,pulsex,pulsey
      common/rkz4/ t,x,v,dt

C
C     pi = 3.14159 26535 89793 23846 26433 83279 50288 41971 69399 37511
C
      pi = 3.14159265358979
C####
C
C     hbar = 1.0545887d-34 J sec
C
      hbar = 1.0545887d-34
C####

C
C     Read parameters in traj.par
C         t1, dt in sec
C         res in Hz
C         nptsft a power of 2
C         narray < 47059
C         x(i) in centimeter
C         vstart(i) in centimeter/sec
C
      open(unit=8,file='traj.par',status='old')
      read(8,1) file1
      read(8,1) file2
 1    format(1x,a60)
      open(unit=9,file=file1,status='new',form='unformatted')
      open(unit=10,file=file2,status='new',form='unformatted')
      read(8,6) t1
      read(8,6) dt
      read(8,6) res
      read(8,2) nptsft
      fnyqst = res*dfloat(nptsft)/2.0
 2    format(1x,i10)
      read(8,3) narray
 3    format(1x,i5)
      do 4 i=1,nvar
```

```
        read(8,6) x(i)
 4  continue
    do 5 i=1,nvar
        read(8,6) v(i)
 5  continue
 6  format(1x,d30.15)
C
C   Set up charge (multiple of e), ionic mass (in kg)
C       Note:    e = 1.6021892D-19 coulomb
C            mass of electron = 9.109534D-31 kg
C            mass of proton = 1.6726485D-27 kg
C            spin = 1. for up, -1. for down (spin-1/2)
C            "magmom" = magnetic moment
C            electron magmom = 9.284832d-17 erg/tesla
C            proton magmom = 1.4106171d-19 erg/tesla
C
    read(8,6) q
    read(8,6) m
    read(8,6) spin
    read(8,6) magmom
C
C   Penning trap: v0 in volt, d in centimeter, b0 in tesla
C
    read(8,6) v0
    read(8,6) d
    read(8,6) b0
C
C   Gradient fields scaled to reach a maximum percentage of the
C       static fields at distance d from the origin.
C       Read ratios for "gradb", "grade"; compute gradients.
C       Units of "gradb" are tesla/centimeter.
C       Units of "grade" are volt/centimeter.
C
    read(8,6) gradb
    gradb = gradb*b0/d
    read(8,6) grade
    grade = grade*v0/d
C
C   Magnetic bottle: b2 in tesla/meter**2 = gauss/cm**2
C
    read(8,6) b2
C
C   Convert b2 to tesla/centimeter**2, as required by DERIVS
C
    b2 = b2*1.d-4
C
C   Parameter for ICR:    cwfreq in Hz
C               cwx,cwy,pulsex,pulsey in V/cm
C
    read(8,6) cwfreq
    read(8,6) cwx
    read(8,6) cwy
    read(8,6) pulsex
    read(8,6) pulsey
C
C   Convert cwfreq to radians/sec
C
    cwfreq = 2.*pi*cwfreq
C
C   Close TRAJ.PAR
C
    close(unit=8)
C####
C
C   Calculate the normal mode frequencies (in the absence of b2)
C       (d/100. is d in m, not cm)
C
    wz = sqrt(q*v0/(m*((d/100.)**2)))
    wc = q*b0/m
    wm = 0.5*wc - 0.5*sqrt(wc*wc - 2*wz*wz)
    wp = wc-wm
C
C   Output frequencies
C
    write(6,*)'vc = ',wc/(2.*pi),' Hz'
    write(6,*)'vp = ',wp/(2.*pi),' Hz'
    write(6,*)'vm = ',wm/(2.*pi),' Hz'
    write(6,*)'vz = ',wz/(2.*pi),' Hz'
C####
C
C   Load starting values
C
    tmin1=sngl(t1)
    t=t1
    write(9) narray
    write(9) sngl(x(1)),sngl(x(2)),sngl(x(3))
    ncount=1
    nft=1
    write(10) nptsft,fnyqst
    write(10) x(1),x(2),x(3),t
C####
C
C   Output initial position, velocity, and quantum numbers
C       (factor of 1.d-4 in quant(n,k,l) needed since
C       x in cm, v in cm/sec, so result is (cm/m)**2;
C       1.d-4 makes the quantum numbers dimensionless)
C
    write(6,*)'t = ',t,' sec'
    write(6,*)'x 1-3:',x(1),x(2),x(3),' cm'
    write(6,*)'v 1-3:',v(1),v(2),v(3),' cm/sec'
    quantn=(m/(2.*hbar*(wp-wm)))*((v(1)+wm*x(2))**2 +
    1        (v(2)-wm*x(1))**2)*1.d-4
    quantl=(m/(2.*hbar*(wp-wm)))*((v(1)+wp*x(2))**2 +
    1        (v(2)-wp*x(1))**2)*1.d-4
    quantk=(m/(2.*hbar))*(wz*x(3)*x(3)+v(3)*v(3)/wz)*1.d-4
    write(6,*)'n,k,l:',quantn,quantk,quantl
C####
C
C   Calculate nperft = number trajectory points per FT point
C
    nperft = nint( 1.0 / ( dt*res*dfloat(nptsft) ) )
    nstep = nperft*(nptsft-1)+1
```

```
      nnext=nft*nperft + 1
C
C     Take first NARRAY steps, filling tt1,traj1
C
      do 13 k=2,narray
           call rkz4
           ncount=ncount+1
           write(9) sngl(x(1)),sngl(x(2)),sngl(x(3))
           if(ncount.eq.nnext) then
                write(10) x(1),x(2),x(3),t
                nft=nft+1
                nnext=nft*nperft+1
           endif
 13   continue
           tmax1=sngl(t)
C
C     Take next NSTEP-2*NARRAY steps
C
      do 113 k=narray+1,nstep-narray
           call rkz4
           ncount=ncount+1
           if(ncount.eq.nnext) then
                write(10) x(1),x(2),x(3),t
                nft=nft+1
                nnext=nft*nperft+1
           endif
 113  continue
C
C     Take final NARRAY steps, filling tt2,traj2
C
      tmin2 = sngl(t+dt)
      do 213 k=1,narray
           call rkz4
           ncount=ncount+1
           write(9) sngl(x(1)),sngl(x(2)),sngl(x(3))
           if(ncount.eq.nnext) then
                write(10) x(1),x(2),x(3),t
                nft=nft+1
                nnext=nft*nperft+1
           endif
 213  continue
           tmax2=sngl(t)

      write(9) tmin1,tmax1,tmin2,tmax2
C
C     Output x,v,t to screen
C
C     write(6,*) 'x  = ',x(1),' cm'
C     write(6,*) 'y  = ',x(2),' cm'
C     write(6,*) 'z  = ',x(3),' cm'
C     write(6,*) 'vx = ',v(1),' cm/sec'
C     write(6,*) 'vy = ',v(2),' cm/sec'
C     write(6,*) 'vz = ',v(3),' cm/sec'
C     write(6,*) 't  = ',t,' sec'
C####
C
```

```
C     Output initial position, velocity, and quantum numbers
C          (factor of 1.d-4 in quant(n,k,l) needed since
C          x in cm, v in cm/sec, so result is (cm/m)**2;
C          1.d-4 makes the quantum numbers dimensionless)
C
      write(6,*)'t = ',t,' sec'
      write(6,*)'x 1-3:',x(1),x(2),x(3),' cm'
      write(6,*)'v 1-3:',v(1),v(2),v(3),' cm/sec'
      quantn=(m/(2.*hbar*(wp-wm)))*((v(1)+wm*x(2))**2 +
     1         (v(2)-wm*x(1))**2)*1.d-4
      quantl=(m/(2.*hbar*(wp-wm)))*((v(1)+wp*x(2))**2 +
     1         (v(2)-wp*x(1))**2)*1.d-4
      quantk=(m/(2.*hbar))*(wz*x(3)*x(3)+v(3)*v(3)/wz)*1.d-4
      write(6,*)'n,k,l:',quantn,quantk,quantl
C####


      close(unit=9)
      close(unit=10)

C
C     Write new version of TRAJ.PAR.
C
C     open(unit=8,file='traj.par',status='new')
C
C     write(8,1) file1
C     write(8,1) file2
C     write(8,6) t
C     write(8,6) dt
C     write(8,6) res
C     write(8,2) nptsft
C     write(8,3) narray
C     write(8,6) x(1)
C     write(8,6) x(2)
C     write(8,6) x(3)
C     write(8,6) v(1)
C     write(8,6) v(2)
C     write(8,6) v(3)
C     write(8,6) q
C     write(8,6) m
C     write(8,6) spin
C     write(8,6) magmom
C     write(8,6) v0
C     write(8,6) d
C     write(8,6) b0
C     write(8,6) gradb*d/b0
C     write(8,6) grade*d/v0
C     write(8,6) b2*1.d4
C     write(8,6) cwfreq/(2.*pi)
C     write(8,6) cwx
C     write(8,6) cwy
C     write(8,6) pulsex
C     write(8,6) pulsey
C
C     close(unit=8)
```

```
C
C   Write new version of TRAJB2.PAR.
C
    open(unit=8,file='trajb2.par',status='old')
    open(unit=9,file='trajb2.par',status='new')

C   Data file name of TRAJB2.FOR is read from TRAJB2.PAR.
    read(8,1) file1
    write(9,1) file1

    read(8,*)
    write(9,6) t

C   End time of TRAJB2.FOR is read from TRAJB2.PAR.
    read(8,6) t
    write(9,6) t

C   Sampling time of TRAJB2.FOR is read from TRAJB2.PAR.
    read(8,6) dt
    write(9,6) dt

C   Array length of TRAJB2.FOR is read from TRAJB2.PAR.
    read(8,3) narray
    write(9,3) narray

    read(8,*)
    read(8,*)
    read(8,*)
    read(8,*)
    read(8,*)
    read(8,*)
    write(9,6) x(1)
    write(9,6) x(2)
    write(9,6) x(3)
    write(9,6) v(1)
    write(9,6) v(2)
    write(9,6) v(3)
    read(8,6) q
    write(9,6) q
    read(8,6) m
    write(9,6) m
    read(8,6) spin
    write(9,6) spin
    read(8,6) magmom
    write(9,6) magmom
    read(8,6) v0
    write(9,6) v0
    read(8,6) d
    write(9,6) d
    read(8,6) b0
    write(9,6) b0
    read(8,6) gradb
    write(9,6) gradb
    read(8,6) grade
    write(9,6) grade

C   Magnetic bottle of TRAJB2.FOR is read from TRAJB2.PAR.
    read(8,6) b2
    write(9,6) b2

    read(8,6) cwfreq
    write(9,6) cwfreq
    read(8,6) cwx
    write(9,6) cwx
    read(8,6) cwy
    write(9,6) cwy
    read(8,6) pulsex
    write(9,6) pulsex
    read(8,6) pulsey
    write(9,6) pulsey

    close(unit=8)
    close(unit=9)

    end


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                           C
C   RKZ4 - Pedro J. Pizarro, 2/17/89                                        C
C                                                                           C
C       Based on J. A. Zonneveld, "Automatic Numerical                      C
C       Integration" (Mathematical Centre Tracts No. 8,                     C
C       Mathematisch Centrum, Amsterdam, 1964),   pp. 32-40.                C
C       Computes one fourth-order, five-point  Runge-Kutta                  C
C       integration step for a second order differential                    C
C       equation with first derivatives.  T is the independent              C
C       variable, X is the dependent variable, V is the                     C
C       first derivative, and DT is the time step.  DERIVS is               C
C       the subroutine which calculates the second derivatives,             C
C       and the other variables are physical parameters needed              C
C       by DERIVS.  New  values are returned in T, X, and V.                C
C                                                                           C
C   PJP - revisions - 7/6/89                                                C
C                                                                           C
C       Added common blocks /ion/ and /rkz4/.                               C
C                                                                           C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC



    subroutine rkz4

    parameter (nvar=3)
    double precision x(nvar),v(nvar),t,dt,k0(nvar),k1(nvar),
    1    k2(nvar),k3(nvar),k4(nvar),dv(nvar),xx(nvar),vv(nvar),
    2    tt,th4dx(nvar),th4dv(nvar),q,m,magmom,spin,v0,d,b0,b2,
    3    gradb,grade,cwfreq,cwx,cwy,pulsex,pulsey

C
C   Common blocks (added on 7/6/89)
C
```

```
      common/ion/ q,m,magmom,spin,v0,d,b0,b2,gradb,grade,cwfreq,
     1    cwx,cwy,pulsex,pulsey
      common/rkz4/ t,x,v,dt

C
C     Calculate k0(i)
C
      tt = t
      do 10 i=1,nvar
           xx(i) = x(i)
           vv(i) = v(i)
   10 continue
      call derivs(tt,vv,dv,xx)
      do 15   i=1,nvar
           k0(i) = dt*dv(i)
   15 continue

C
C     Calculate k1(i)
C
      tt = t +  dt/2.
      do  20 i=1,nvar
           xx(i) = x(i) + dt*(4.*v(i) + k0(i))/8.
           vv(i) = v(i) + k0(i)/2.
   20 continue
      call derivs(tt,vv,dv,xx)
      do 25   i=1,nvar
           k1(i) = dt*dv(i)
   25 continue

C
C     Calculate  k2(i)
C
      do 30 i=1,nvar
           vv(i) = v(i) +  k1(i)/2.
   30 continue
      call derivs(tt,vv,dv,xx)
      do 35   i=1,nvar
           k2(i) = dt*dv(i)
   35 continue

C
C     Calculate  k3(i)
C
      tt= t + dt
      do 40 i=1,nvar
           xx(i) = x(i) + dt*(2.*v(i) + k2(i))/2.
           vv(i) = v(i) + k2(i)
   40 continue
      call derivs(tt,vv,dv,xx)
      do 45   i=1,nvar
           k3(i) = dt*dv(i)
   45 continue

C
C     If desired, calculate error terms
C
C     tt = t + dt*0.75
C     do 50 i=1,nvar
C          xx(i)=x(i)+dt*(48.*v(i)+7.*k0(i)+11.*k1(i))/64.
C          vv(i)=v(i)+(5.*k0(i)+7.*k1(i)+13.*k2(i)-k3(i))/32.
C50   continue
C     call derivs(tt,vv,dv,xx)
C     do 55   i=1,nvar
C          k4(i) = dt*dv(i)
C55   continue
C     do 57 i=1,nvar
C          th4dx(i)=dt*(2.*k0(i)-3.*(k1(i)+k2(i))+4.*k3(i))/9.
C          th4dv(i)=(2./3.)*(-1.*k0(i)+3.*(k1(i)+k2(i)+k3(i))
C     1         -8.*k4(i))
C57   continue

C
C     Calculate new T, X, V
C
      t = t + dt
      do 60 i=1,nvar
           x(i) = x(i) + dt*(v(i) + (k0(i)+k1(i)+k2(i))/6.)
           v(i) = v(i) + (k0(i) + 2*(k1(i)+k2(i)) + k3(i) )/6.
   60 continue


      return
      end



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                             C
C     DERIVS            Pedro J. Pizarro, 3/2/89                            C
C                                                                             C
C          Sets up the differential equations governing ion                   C
C          trajectories.  Edit this file to change electric                   C
C          and magnetic fields, as well as charge and ionic                   C
C          mass, in a given problem.                                          C
C                                                                             C
C     PJP - revisions - 7/6/89                                                C
C                                                                             C
C          Added common block /ion/.                                          C
C                                                                             C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine derivs(t,v,dvdt,x)

      parameter (n=3)
      double precision t,q,m,spin,v(n),dvdt(n),x(n),e(n),b(n),
     1    v0,d,v0d,gradb,grade,magmom,bmag,b0,b2,rad2,spnfac,
     2    cwfreq,cwx,cwy,pulsex,pulsey

C
C     Common block (added on 7/6/89)
C
```

```
      common/ion/ q,m,magmom,spin,v0,d,b0,b2,gradb,grade,cwfreq,
     1    cwx,cwy,pulsex,pulsey

C
C     Calculate square of transverse radius, rad2 (in cm**2)
C
      rad2 = x(1)*x(1) + x(2)*x(2)
C
C     Set up charge (multiple of e), ionic mass (in kg)
C          Note:    e = 1.6021892D-19 coulomb
C               mass of electron = 9.109534D-31 kg
C               mass of proton = 1.6726485D-27 kg
C               spin = 1. for up, -1. for down (spin-1/2)
C               "magmom" = magnetic moment
C               electron magmom = 9.284832d-17 erg/tesla
C               proton magmom = 1.4106171d-19 erg/tesla
C
C     Penning trap: v0 in volt, d in centimeter, b0 in tesla
C
      v0d = v0/(d*d)
C
C     Gradient fields scaled to reach a maximum percentage of the
C          static fields at distance d from the origin.
C          Units of "gradb" are tesla/centimeter.
C          Units of "grade" are volt/centimeter.
C
C     Set up fields: E (in volt/centimeter) and B (in tesla)
C          b(3) >> b(1),b(2)
C          b2 in tesla/centimeter**2
C
      b(1) = -0.5*gradb*x(1) - b2*x(1)*x(3)
      b(2) = -0.5*gradb*x(2) - b2*x(2)*x(3)
      b(3) = b0 + gradb*x(3) + b2*(x(3)**2-0.5*rad2)
      bmag = dsqrt(b(1)**2 + b(2)**2 + b(3)**2)
      e(1) = 0.5*v0d*x(1) + cwx*dcos(cwfreq*t) + pulsex
      e(2) = 0.5*v0d*x(2) + cwy*dcos(cwfreq*t) + pulsey
      e(3) = -1.0*v0d*x(3) - grade
C
C     Define spnfac
C
      spnfac = (1.0d-3)*spin*magmom/(m*bmag)
C
C     Calculate dv/dt (in centimeter/sec^2) in each direction
C
      dvdt(1) = (q/m)*(e(1)*1.0d4 + b(3)*v(2) - b(2)*v(3))
     1    + spnfac*(gradb**2)*x(1)/4.
     2    + spnfac*b2*x(1)*(-1.*b0 + 0.5*b2*rad2)
      dvdt(2) = (q/m)*(e(2)*1.0d4 + b(1)*v(3) - b(3)*v(1))
     1    + spnfac*(gradb**2)*x(2)/4.
     2    + spnfac*b2*x(2)*(-1.*b0 + 0.5*b2*rad2)
      dvdt(3) = (q/m)*(e(3)*1.0d4 + b(2)*v(1) - b(1)*v(2))
     1    + spnfac*b(3)*gradb
     2    + 2.*spnfac*b2*x(3)*(b0 + b2*x(3)*x(3))

      return
      end
```

A.3 TRAJ_FT.FOR, written for VAX/VMS systems

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                              C
C     TRAJ_FT - Pedro J. Pizarro, 1/27/89                                      C
C                                                                              C
C          Reads the files TRAJ.DAT and TRAJ_FT.DAT created by                 C
C          TRAJ.FOR.  Plots the REAL*4 arrays in TRAJ.DAT in a                 C
C          choice of formats.  Processes the REAL*8 data in                    C
C          TRAJ_FT.DAT by referencing to a specified cosine                    C
C          (input interactively) and taking the Fast Fourier                   C
C          Transform of the resulting waveform.  This is then                  C
C          plotted on the screen, or saved as REAL*8 in the                    C
C          unformatted file TRAJ_FT.FFT.                                       C
C                                                                              C
C     PJP - Revisions - 7/7/89                                                 C
C                                                                              C
C          Increased narrmx from 100000 to 140000, in order to                 C
C          process arrays with 65536 points.  Added error warnings             C
C          for larger arrays.                                                  C
C     C
C          Added threshold control to peak finder for FFT's  (a la             C
C          ZEROFILL.FOR).                                                      C
C     C
C                                                                              C
C     PJP - Revisions - 7/10/89                                                C
C                                                                              C
C          Can now damp the signal to be Fourier transformed.                  C
C                                                                              C
C     PJP - Revisions - 7/12/89                                                C
C                                                                              C
C          Made plotting of x, y, or z vs. t optional.                         C
C                                                                              C
C     PJP - Revisions - 9/1/89                                                 C
C                                                                              C
C          Changed input format of scaling factors for plots to *.             C
C                                                                              C
C     PJP - Revisions - 9/5/89                                                 C
C                                                                              C
C          Added xmax, xmin, ymax, ymin, zmax, zmin: they are                  C
C          output to the screen before scale factors for time                  C
C          plots are chosen.                                                   C
C                                                                              C
C     PJP - Revisions - 5/18/90                                                C
C                                                                              C
C          Changed FFT routines - now do complex transform, with               C
```

```
C       menu-driven format.                                          C
C                                                                    C
C   Revisions - Pedro J. Pizarro, 6/13/90
    C
C                                                                    C
C       Now accepts the terminal label 't4010' for PC emulation      C
C       of Tek4010 terminals.                                        C
C                                                                    C
C   Revisions - Pedro J. Pizarro, 12/6/90
    C
C                                                                    C
C       Now gives option of producing time-domain output files       C
C       to be plotted by PLT2.
    C
C                                                                    C
C   Revisions - Pedro J. Pizarro, 4/30/91
    C
C                                                                    C
C       Eliminated zero padding between initial and final time       C
C       snapshots in time domain plots.
    C
C                                                                    C
C   Revisions - Pedro J. Pizarro, 5/2/91
    C
C                                                                    C
C       Damping for FFT files is now specified as a linewidth        C
C       (FWHM) in Hz.                                                C
C                                                                    C
C   Revisions - Pedro J. Pizarro, 10/29/91
    C
C                                                                    C
C       Option to write time domain files for PLT2 now asks          C
C       for IKEEP; every IKEEP point is kept.  Thus, points          C
C       1,1+IKEEP,1+2*IKEEP,... are kept.
    C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      program traj_ft

      parameter (narrmx=132000)
      real*8 fnyqst,xpos(narrmx),ypos(narrmx),zpos(narrmx),time(narrmx),
     1   freq,ftreal(narrmx),ftimag(narrmx),
     2   res,freqcy(narrmx),phase,ftmax,pi,damp,
     3   sgreal(narrmx),sgimag(narrmx),ftmagn(narrmx)
      real*4 xplot(narrmx),yplot(narrmx),ttplot(narrmx),xtplot(narrmx),
     1   ytplot(narrmx),ztplot(narrmx),tmin1,tmax1,tmin2,tmax2,
     2   ftmax2,xmax,xmin,ymax,ymin,zmax,zmin,
     3   pltftr(narrmx),pltfti(narrmx),pltftm(narrmx),pltfrq(narrmx)
      integer narray,nptsft,ntplot,nfplot,iterm,ikeep
      character*6 answer
      character*5 terminal
      character*9 filename
      character*1 answr2,answr3
      character*60 file1,file2
C
C     pi = 3.14159 26535 89793 23846 26433 83279 50288 41971 69399 37511
C
      pi = 3.141592653589793
C
C     Get terminal type
C
 4    write(6,5)
 5    format(5x,'Terminal:  1 = VT240  2 = LANPAR  3 = TEK4010 ',
     1   'emulation (PC)')
      read(5,*) iterm
 6    format(a5)
      if( iterm.ne.1 .and. iterm.ne.2 .and. iterm.ne.3) goto 4
      if(iterm.eq.1) terminal = 'vt240'
      if(iterm.eq.2) terminal = 'lanpa'
      if(iterm.eq.3) terminal = 't4010'
C
C     Get data file names
C
      write(6,7)
 7    format(5x,'Enter name of trajectory snapshot data file, or NONE:')
      read(5,9) file1
      write(6,8)
 8    format(5x,'Enter name of FT time domain data file, or NONE:')
      read(5,9) file2
 9    format(a60)
C
C     Load data for time-domain plots
C
      if(file1.eq.'none'.or.file1.eq.'NONE') goto 1000
      open(unit=9,file=file1,status='old',form='unformatted')
      read(9) narray
C####     ntplot=ifix(float(narray)*85./40.)
      ntplot=narray*2
      if(ntplot.gt.narrmx) then
          ntplot=narrmx
          write(6,*) 'ERROR WARNING: Too many points for time plots'
      endif
C
C     Take first NARRAY steps
C
      read(9)xplot(1),yplot(1),ztplot(1)
      xtplot(1)=xplot(1)
      ytplot(1)=yplot(1)
      xmax=xtplot(1)
      xmin=xtplot(1)
      ymax=ytplot(1)
      ymin=ytplot(1)
      zmax=ztplot(1)
      zmin=ztplot(1)

      do 13 k=2,narray
          read(9)xplot(k),yplot(k),ztplot(k)
          xtplot(k)=xplot(k)
          ytplot(k)=yplot(k)
          xmax=amax1(xmax,xtplot(k))
          xmin=amin1(xmin,xtplot(k))
```

```
        ymax=amax1(ymax,ytplot(k))
        ymin=amin1(ymin,ytplot(k))
        zmax=amax1(zmax,ztplot(k))
        zmin=amin1(zmin,ztplot(k))
 13  continue
C
C   Zero-fill time plot records
C
C####    do 150 k=narray+1,ntplot-narray
C####        xtplot(k)=0.
C####        ytplot(k)=0.
C####        ztplot(k)=0.
C#### 150    continue
C
C   Take final NARRAY steps
C
    do 213 k=1,narray
        read(9) xplot(narray+k),yplot(narray+k),
    1       ztplot(ntplot-narray+k)
        xtplot(ntplot-narray+k)=xplot(narray+k)
        ytplot(ntplot-narray+k)=yplot(narray+k)
        xmax=amax1(xmax,xtplot(ntplot-narray+k))
        xmin=amin1(xmin,xtplot(ntplot-narray+k))
        ymax=amax1(ymax,ytplot(ntplot-narray+k))
        ymin=amin1(ymin,ytplot(ntplot-narray+k))
        zmax=amax1(zmax,ztplot(ntplot-narray+k))
        zmin=amin1(zmin,ztplot(ntplot-narray+k))
 213 continue
C
C   Read time extrema
C
    read(9) tmin1,tmax1,tmin2,tmax2
    close(unit=9)
C
C   Load data for FFT
C
1000    if(file2.eq.'none'.or.file2.eq.'NONE') goto 2000
    open(unit=10,file=file2,status='old',form='unformatted')
    read(10) nptsft,fnyqst
    nfplot=ifix(float(nptsft/2+1)*85./40.)
    if(nfplot.gt.narrmx) then
        nfplot=narrmx
        write(6,*) 'ERROR WARNING: Too many points for frequency plots'
    endif
    do 250 i=1,nptsft
        read(10) xpos(i),ypos(i),zpos(i),time(i)
 250 continue
    close(unit=10)
C
C   Write data index array (used for time and frequency plots)
C
 2000    do 214 i=1,max(ntplot,nfplot)
        ttplot(i)=float(i)
 214 continue
C
C   Plot time domain, or take FFT?
C
  40 write(6,42)
  42 format(5x,'Enter PLOTXY,PLOTXT,PLOTYT,PLOTZT,LINEXY,LINEXT,
     1 LINEYT,')
     write(6,43)
  43 format(11x,'LINEZT,PLOTFX,LINEFX,PLOTFY,LINEFY,PLOTFZ,LINEFZ,
     1 or EXIT:')
     read(5,44)answer
  44 format(a6)
     if(answer.eq.'plotxy' .or. answer.eq.'PLOTXY') goto 300
     if(answer.eq.'linexy' .or. answer.eq.'LINEXY') goto 300
     if(answer.eq.'plotxt' .or. answer.eq.'PLOTXT') goto 310
     if(answer.eq.'linext' .or. answer.eq.'LINEXT') goto 310
     if(answer.eq.'plotyt' .or. answer.eq.'PLOTYT') goto 320
     if(answer.eq.'lineyt' .or. answer.eq.'LINEYT') goto 320
     if(answer.eq.'plotzt' .or. answer.eq.'PLOTZT') goto 330
     if(answer.eq.'linezt' .or. answer.eq.'LINEZT') goto 330
     if(answer.eq.'plotfx' .or. answer.eq.'PLOTFX') goto 400
     if(answer.eq.'linefx' .or. answer.eq.'LINEFX') goto 400
     if(answer.eq.'plotfy' .or. answer.eq.'PLOTFY') goto 400
     if(answer.eq.'linefy' .or. answer.eq.'LINEFY') goto 400
     if(answer.eq.'plotfz' .or. answer.eq.'PLOTFZ') goto 400
     if(answer.eq.'linefz' .or. answer.eq.'LINEFZ') goto 400
     if(answer.eq.'exit' .or. answer.eq.'EXIT') goto 500
     goto 40
C
C   If unavailable file called:
C
 5000    write(6,*)
     write(6,*) '    Required data file not available.'
     write(6,*)
     goto 40
C
C   Plot trajectory
C
 300 if(file1.eq.'none'.or.file1.eq.'NONE') goto 5000
     write(6,*)'xmax = ',xmax,' xmin = ',xmin,' cm'
     write(6,*)'ymax = ',ymax,' ymin = ',ymin,' cm'
 302 write(6,*)'    Plot trajectory?   Y/N:'
     read(5,474,err=302)answr2
     if(answr2.eq.'n' .or. answr2.eq.'N') goto 304
     if(answr2.ne.'y' .and. answr2.ne.'Y') goto 302
     call plot(narrmx,answer,narray*2,xplot,yplot,terminal)
C
C   Output trajectory for PLT2 plotting?
C
 304 write(6,*)'    Write x-y plot to file XY.DAT for plotting by',
     1   ' PLT2?  Y/N:'
     read(5,474,err=304)answr2
     if(answr2.eq.'n' .or. answr2.eq.'N') goto 40
     if(answr2.ne.'y' .and. answr2.ne.'Y') goto 304
 305 write(6,*)'    Keep every Nth point; enter N (1 to keep all):'
     read(5,*,err=305) ikeep
     open(unit=8,file='xy.dat',status='new')
     do 306 i=1,2*narray,ikeep
         write(8,*) xplot(i),yplot(i)
```

```
 306 continue
     close(unit=8)
     goto 40

 310 if(file1.eq.'none'.or.file1.eq.'NONE') goto 5000
     write(6,*)'xmax = ',xmax,' xmin = ',xmin,' cm'
 3100    write(6,*)
     write(6,*)'     Plot this trajectory record? y/n'
     write(6,*)
     read(5,474) answr2
 474 format(a1)
     if(answr2.eq.'y' .or. answr2.eq.'Y') goto 3101
     if(answr2.eq.'n' .or. answr2.eq.'N') goto 311
     goto 3100
 3101    call timeplot(narrmx,ntplot,answer,narray*2,ttplot,xtplot,
     1   tmin1,tmax1,tmin2,tmax2,terminal)
 311     write(6,*)
     write(6,*)'     Find peaks? y/n'
     write(6,*)
     read(5,474) answr2
     if(answr2.eq.'y' .or. answr2.eq.'Y') goto 312
     if(answr2.eq.'n' .or. answr2.eq.'N') goto 314
     goto 311
 312 write(6,*)'      ***** Peaks in x motion *****'
     call peak2(narrmx,narray,ntplot,ttplot,xtplot,tmin1,tmax1,
     1   tmin2,tmax2)
C
C    Output trajectory for PLT2 plotting?
C
 314 write(6,*)'    Write x-t plot to file XT.DAT for plotting by',
     1   ' PLT2?  Y/N:'
     read(5,474,err=314)answr2
     if(answr2.eq.'n' .or. answr2.eq.'N') goto 40
     if(answr2.ne.'y' .and. answr2.ne.'Y') goto 314
 315 write(6,*)'    Keep every Nth point; enter N (1 to keep all):'
     read(5,*,err=315) ikeep
     open(unit=8,file='xt.dat',status='new')
     do 316 i=1,ntplot,ikeep
         write(8,*) ttplot(i),xtplot(i)
 316 continue
     close(unit=8)
     goto 40

 320 if(file1.eq.'none'.or.file1.eq.'NONE') goto 5000
     write(6,*)'ymax = ',ymax,' ymin = ',ymin,' cm'
 3200    write(6,*)
     write(6,*)'     Plot this trajectory record? y/n'
     write(6,*)
     read(5,474) answr2
     if(answr2.eq.'y' .or. answr2.eq.'Y') goto 3201
     if(answr2.eq.'n' .or. answr2.eq.'N') goto 321
     goto 3200
 3201    call timeplot(narrmx,ntplot,answer,narray*2,ttplot,ytplot,
     1   tmin1,tmax1,tmin2,tmax2,terminal)
 321     write(6,*)
     write(6,*)'     Find peaks? y/n'
     write(6,*)
     read(5,474) answr2
     if(answr2.eq.'y' .or. answr2.eq.'Y') goto 322
     if(answr2.eq.'n' .or. answr2.eq.'N') goto 324
     goto 321
 322 write(6,*)'      ***** Peaks in y motion *****'
     call peak2(narrmx,narray,ntplot,ttplot,ytplot,tmin1,tmax1,
     1   tmin2,tmax2)
C
C    Output trajectory for PLT2 plotting?
C
 324 write(6,*)'    Write y-t plot to file YT.DAT for plotting by',
     1   ' PLT2?  Y/N:'
     read(5,474,err=324)answr2
     if(answr2.eq.'n' .or. answr2.eq.'N') goto 40
     if(answr2.ne.'y' .and. answr2.ne.'Y') goto 324
 325 write(6,*)'    Keep every Nth point; enter N (1 to keep all):'
     read(5,*,err=325) ikeep
     open(unit=8,file='yt.dat',status='new')
     do 326 i=1,ntplot,ikeep
         write(8,*) ttplot(i),ytplot(i)
 326 continue
     close(unit=8)
     goto 40

 330 if(file1.eq.'none'.or.file1.eq.'NONE') goto 5000
     write(6,*)'zmax = ',zmax,' zmin = ',zmin,' cm'
 3300    write(6,*)
     write(6,*)'     Plot this trajectory record? y/n'
     write(6,*)
     read(5,474) answr2
     if(answr2.eq.'y' .or. answr2.eq.'Y') goto 3301
     if(answr2.eq.'n' .or. answr2.eq.'N') goto 331
     goto 3300
 3301    call timeplot(narrmx,ntplot,answer,narray*2,ttplot,ztplot,
     1   tmin1,tmax1,tmin2,tmax2,terminal)
 331     write(6,*)
     write(6,*)'     Find peaks? y/n'
     write(6,*)
     read(5,474) answr2
     if(answr2.eq.'y' .or. answr2.eq.'Y') goto 332
     if(answr2.eq.'n' .or. answr2.eq.'N') goto 334
     goto 331
 332 write(6,*)'      ***** Peaks in axial (z) motion *****'
     call peak2(narrmx,narray,ntplot,ttplot,ztplot,tmin1,tmax1,
     1   tmin2,tmax2)
C
C    Output trajectory for PLT2 plotting?
C
 334 write(6,*)'    Write z-t plot to file ZT.DAT for plotting by',
     1   ' PLT2?  Y/N:'
     read(5,474,err=334)answr2
     if(answr2.eq.'n' .or. answr2.eq.'N') goto 40
     if(answr2.ne.'y' .and. answr2.ne.'Y') goto 334
 335 write(6,*)'    Keep every Nth point; enter N (1 to keep all):'
     read(5,*,err=335) ikeep
```

```
      open(unit=8,file='zt.dat',status='new')
      do 336 i=1,ntplot,ikeep
          write(8,*) ttplot(i),ztplot(i)
  336 continue
      close(unit=8)
      goto 40

   70 write(6,*) ' --->> FFT?    Yes - 1; No - 0'
      write(6,*)
      read(5,*,err=70) i
      if(i.ne.0 .and. i.ne.1) goto 70
      if(i.eq.0) goto 40

C
C     FFT - Ask for reference frequency; multiply data by
C         exp(2*pi*I*reference*t); take FFT; plot; save or not
C
  400 if(file2.eq.'none'.or.file2.eq.'NONE') goto 5000
      isign = -1
      write(6,*) '* * * * * * * * * * F F T * * * * * * * * * * '
      write(6,*)
20000     write(6,20010)
20010     format(5x,'Enter reference frequency, in Hz:')
      read(5,*,err=20000) freq

20015     write(6,20016)
20016     format(5x,'Enter reference phase, in degrees:')
      read(5,*,err=20015) phase
      phase = phase * pi / 180.

C20018    write(6,20019)
C20019    format(5x,'Enter the damping time constant, in 1/sec:')
C     read(5,*,err=20018) damp
20018     write(6,20019)
20019     format(5x,'Enter the damping linewidth, in Hz:')
      read(5,*,err=20018) damp
      damp = pi*dabs(damp)   !Convert damp to 1/sec (1/T2), >= 0

      if (answer.eq.'plotfx' .or. answer.eq.'PLOTFX') goto 430
      if (answer.eq.'linefx' .or. answer.eq.'LINEFX') goto 430
      if (answer.eq.'plotfy' .or. answer.eq.'PLOTFY') goto 440
      if (answer.eq.'linefy' .or. answer.eq.'LINEFY') goto 440
      if (answer.eq.'plotfz' .or. answer.eq.'PLOTFZ') goto 450
      if (answer.eq.'linefz' .or. answer.eq.'LINEFZ') goto 450
  430 filename = 'fftx.dat'
      do 435 i=1,nptsft
          sgreal(i) = xpos(i)
          sgimag(i) = 0.
  435 continue
      goto 460

  440 filename = 'ffty.dat'
      do 445 i=1,nptsft
          sgreal(i) = ypos(i)
          sgimag(i) = 0.
  445 continue
      goto 460

  450 filename = 'fftz.dat'
      do 455 i=1,nptsft
          sgreal(i) = zpos(i)
          sgimag(i) = 0.
  455 continue

  460 do 20030 i=1,nptsft
          cc = cos(2.*pi*freq*time(i) + phase)
     1         *exp(-1.*time(i)*damp)
          ss = sin(2.*pi*freq*time(i) + phase)
     2         *exp(-1.*time(i)*damp)
          ftreal(i) = sgreal(i)*cc - sgimag(i)*ss
          ftimag(i) = sgreal(i)*ss + sgimag(i)*cc
20030     continue

20032     write(6,*) 'NPTS=',nptsft,';  Zerofill?   Yes - 1 ; No - 0'
      read(5,*,err=20032) i
      write(6,*)
      if(i.ne.0 .and. i.ne.1) goto 20032
      if(i.eq.0) then
          npttot = nptsft
          goto 20035
      else
20033         write(6,*) 'Enter total number of points',
     1         ' (multiple of 2):'
          read(5,*,err=20033) npttot
          write(6,*)
          if(npttot.le.nptsft) goto 20032
          do 20034 j=nptsft+1,npttot
              ftreal(j)=0.
              ftimag(j)=0.
20034         continue
      endif


20035     call fft_complex(ftreal,ftimag,npttot,ftreal,ftimag,isign)

C
C     Calculate actual frequencies
C
      res = 2.0*fnyqst/dfloat(npttot)

      do 20037 i=1,npttot
          freqcy(i) = -1.*fnyqst + res*float(i)
          pltfrq(i) = sngl(freqcy(i))
20037     continue

      ftrmx = 0.
      ftrmn = 0.
      ftimx = 0.
      ftimn = 0.
      ftmmx = 0.
      ftmmn = 0.
```

```
      do 20040 i=1,npttot
         ftmagn(i) = sqrt(ftreal(i)**2 + ftimag(i)**2)
         pltftr(i) = sngl(ftreal(i))
         pltfti(i) = sngl(ftimag(i))
         pltftm(i) = sngl(ftmagn(i))
         ftrmx = amax1(ftrmx,pltftr(i))
         ftrmn = amin1(ftrmn,pltftr(i))
         ftimx = amax1(ftimx,pltfti(i))
         ftimn = amin1(ftimn,pltfti(i))
         ftmmx = amax1(ftmmx,pltftm(i))
         ftmmn = amin1(ftmmn,pltftm(i))
20040    continue

20050    if(isign.eq.1) write(6,20060)
      if(isign.eq.-1) write(6,20061)
      write(6,20064)
      write(6,20066)
      write(6,20068)
      write(6,20070)
      write(6,20072)
      write(6,20074)
      write(6,20076)
      write(6,20077)
      write(6,20078)

20060    format(5x,'FFT - ISIGN = 1 --->> Options:')
20061    format(5x,'FFT - ISIGN = -1 --->> Options:')
20064    format(7x,'1 - Plot real part')
20066    format(7x,'2 - Plot imaginary part')
20068   format(7x,'3 - Plot magnitude')
20070    format(7x,'4 - Find peaks in real part')
20072    format(7x,'5 - Find peaks in imaginary part')
20074    format(7x,'6 - Find peaks in magnitude')
20076    format(7x,'7 - Change FFT parameters')
20077    format(7x,'8 - Output FFT to files for plotting by PLT2')
20078    format(7x,'9 - Return to main program')

      read(5,*,err=20050) ianswr

      if(ianswr.eq.1) goto 20110
      if(ianswr.eq.2) goto 20120
      if(ianswr.eq.3) goto 20130
      if(ianswr.eq.4) goto 20140
      if(ianswr.eq.5) goto 20150
      if(ianswr.eq.6) goto 20160
      if(ianswr.eq.7) goto 20000
      if(ianswr.eq.8) goto 20080
      if(ianswr.eq.9) goto 40
      goto 20050

C
C     Output FFT for PLT2 plotting
C
20080    open(unit=8,file='ftreal.dat',status='new')
      open(unit=9,file='ftimag.dat',status='new')
      open(unit=10,file='ftmagn.dat',status='new')

      do 20082 i=1,npttot
         write(8,*) pltfrq(i),pltftr(i)
         write(9,*) pltfrq(i),pltfti(i)
         write(10,*) pltfrq(i),pltftm(i)
20082    continue

      close(unit=8)
      close(unit=9)
      close(unit=10)

      goto 20050

20110    call plotf(narrmx,'line',npttot,pltfrq,pltftr,terminal,
     1   pltfrq(npttot),pltfrq(1),ftrmx,ftrmn)
      write(6,*) '    Nyquist frequency = ',fnyqst,' Hz'
      write(6,*) '          min = ',ftrmn,'  max = ',ftrmx
      write(6,*)
      goto 20050

20120    call plotf(narrmx,'line',npttot,pltfrq,pltfti,terminal,
     1   pltfrq(npttot),pltfrq(1),ftimx,ftimn)
      write(6,*) '    Nyquist frequency = ',fnyqst,' Hz'
      write(6,*) '          min = ',ftimn,'  max = ',ftimx
      write(6,*)
      goto 20050

20130    call plotf(narrmx,'line',npttot,pltfrq,pltftm,terminal,
     1   pltfrq(npttot),pltfrq(1),ftmmx,ftmmn)
      write(6,*) '    Nyquist frequency = ',fnyqst,' Hz'
      write(6,*) '          min = ',ftmmn,'  max = ',ftmmx
      write(6,*)
      goto 20050

20140    ftmax = dble(amax1(abs(ftrmx),abs(ftrmn)))
      call peakf(narrmx,npttot,fnyqst,freq,freqcy,ftreal,ftmax)
      goto 20050

20150    ftmax = dble(amax1(abs(ftimx),abs(ftimn)))
      call peakf(narrmx,npttot,fnyqst,freq,freqcy,ftimag,ftmax)
      goto 20050

20160    ftmax = dble(amax1(abs(ftmmx),abs(ftmmn)))
      call peakf(narrmx,npttot,fnyqst,freq,freqcy,ftmagn,ftmax)
      goto 20050


  500 end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C    PLOT - Pedro J. Pizarro, 1/18/89
     C
C         Uses GKS to plot on a VT240 or a Lanpar Vision II 3222.      C
C                                                                      C
```

```
C    Revisions - Pedro J. Pizarro, 9/1/89
     C
C                                                                       C
C        Now adjusts order for borders larger than 1.e7.                C
C                                                                       C
C    Revisions - Pedro J. Pizarro, 6/13/90
     C
C                                                                       C
C        Now accepts the terminal label 't4010' for PC emulation        C
C        of Tek4010 terminals.                                          C
C                                                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

     subroutine plot(narrmx,answer,npts,xplot,yplot,terminal)
     real*4 xbordr,ybordr,xplot(narrmx),yplot(narrmx),xpt(4),
     1    ypt(4),zero(4),x(20000),y(20000),xmin,xmax,ymin,ymax,
     2    xm(3),ym(3),scale,xori,yori,order
     integer npts,nunits,nrmndr
     character*4 answer
     character*5 terminal
     character*10 cxpt(3),cypt(3)
     zero(1)=0.
     zero(2)=0.
     zero(3)=0.
     zero(4)=0.

     write(6,100)
 100 format(5x,'Enter window scale:')
     read(5,*)scale
     write(6,110)
 110 format(5x,'Enter horizontal center of plot:')
     read(5,*)xori
     write(6,120)
 120 format(5x,'Enter vertical center of plot:')
     read(5,*)yori

     xmax=xori+scale
     xmin=xori-scale
     ymax=yori+scale
     ymin=yori-scale
C
C    Since GKS does not like window borders smaller than ~1.e-4,
C    or larger that ~1.e8, and since the scales involved here are
C    smaller than that, adjust xmax, xmin, ymax, and ymin appropriately
C



     order=1.
 300 if((abs(xmax).lt.1.e-4 .and. xmax.ne.0.) .or.
     1    (abs(xmin).lt.1.e-4 .and. xmin.ne.0.) .or.
     2    (abs(ymax).lt.1.e-4 .and. ymax.ne.0.) .or.
     3    (abs(ymin).lt.1.e-4 .and. ymin.ne.0.)) goto 310

     goto 350

 310 order=order*10.
     xmax=xmax*10.
     xmin=xmin*10.
     ymax=ymax*10.
     ymin=ymin*10.
     goto 300

 350 if((abs(ymax).gt.1.e7) .or. (abs(ymin).gt.1.e7) .or.
     1    (abs(xmax).gt.1.e7) .or. (abs(xmin).gt.1.e7))
     2    goto 360

     goto 400

 360 order=order/10.
     xmax=xmax/10.
     xmin=xmin/10.
     ymax=ymax/10.
     ymin=ymin/10.
     goto 350
C
C    Set up borders
C
 400 xbordr=(xmax-xmin)*7.5/85.
     ybordr=(ymax-ymin)*7.5/85.

     xpt(1)=xmin
     xpt(2)=xmax
     xpt(3)=xmin + (xmax-xmin)/2
     ypt(1)=ymin
     ypt(2)=ymax
     ypt(3)=ymin + (ymax-ymin)/2
     xm(1)=xmin
     xm(2)=xmin
     xm(3)=xmin
     ym(1)=ymin
     ym(2)=ymin
     ym(3)=ymin
C
C
C    Create axis labels
C
     open(unit=11,file='trajscr.tch',status='new')
     do 40 i=1,3
        write(11,42)xpt(i)/order,ypt(i)/order
 40  continue
 42  format(1x,e10.2,1x,e10.2)
     close(unit=11,status='keep')

     open(unit=12,file='trajscr.tch',status='old')
     do 44 i=1,3
        read(12,46)cxpt(i),cypt(i)
 44  continue
 46  format(1x,a10,1x,a10)
     close(unit=12,status='delete')

C
C    Initialize GKS
```

```
C
      if((terminal.eq.'lanpa').or.(terminal.eq.'LANPA')) then
          call lanpar_graph
          call gks$open_gks('sys$error:')
          call gks$open_ws(1,'tta1:',72)
      endif
      if(terminal.eq.'vt240') then
          call gks$open_gks('sys$error:')
          call gks$open_ws(1,0,0)
      endif
      if(terminal.eq.'t4010') then
          call gks$open_gks('sys$error:')
          call gks$open_ws(1,'sys$output:',72)
      endif
      call gks$activate_ws(1)
C
C     Set graphics window size
C
      call gks$set_window(1,xmin-xbordr,xmax+xbordr,
     1    ymin-ybordr,ymax+ybordr)
      call gks$select_xform(1)
C
C     Draw axes
C
      call gks$polyline(2,xpt,ym)
      call gks$polyline(2,xm,ypt)
      if(xmax.ge.0. .and. xmin.le.0) call gks$polyline(2,zero,ypt)
      if(ymax.ge.0. .and. ymin.le.0) call gks$polyline(2,xpt,zero)
C
C     Draw ticks, write labels
C
      call gks$set_pmark_size(1.0)
      call gks$set_pmark_type(2)
      call gks$polymarker(3,xpt,ym)
      call gks$set_text_height(0.015*(2*ybordr+ymax-ymin))
      call gks$set_text_align(2,1)
      call gks$text(xpt(1),ymin,cxpt(1))
      call gks$text(xpt(2),ymin,cxpt(2))
      call gks$text(xpt(3),ymin,cxpt(3))
      call gks$polymarker(3,xm,ypt)
      call gks$set_text_align(2,5)
      call gks$text(xmin,ypt(1),cypt(1))
      call gks$text(xmin,ypt(2),cypt(2))
      call gks$text(xmin,ypt(3),cypt(3))
      call gks$set_pmark_type(1)

      nunits = ifix(float(npts)/20000.)
      nrmndr = mod(npts,20000)

      if(answer.eq.'line' .or. answer.eq.'LINE') goto 15

      if(nunits.eq.0)goto 54

      do 52 j=1,nunits
          do 50 i=1,20000
              index=(j-1)*20000 + i
              x(i)=xplot(index)*order
              y(i)=yplot(index)*order
50        continue
          call gks$polymarker(20000,x,y)
52        continue

          if(nrmndr.eq.0)goto 19

54    do 56 i=1,nrmndr
          index=nunits*20000+i
          x(i)=xplot(index)*order
          y(i)=yplot(index)*order
56        continue
          call gks$polymarker(nrmndr,x,y)
      goto 19

15    if(nunits.eq.0)goto 64

      do 62 j=1,nunits
          do 60 i=1,20000
              index=(j-1)*20000 + i
              x(i)=xplot(index)*order
              y(i)=yplot(index)*order
60        continue
          call gks$polyline(20000,x,y)
62    continue

          if(nrmndr.eq.0)goto 19
64    do 66 i=1,nrmndr
          index=nunits*20000+i
          x(i)=xplot(index)*order
          y(i)=yplot(index)*order
66        continue
          call gks$polyline(nrmndr,x,y)

19    write(6,20)
20    format(5x,'Press <RETURN> to close this plot:')
      read(5,25)
25    format(1x)
      call gks$deactivate_ws(1)
      call gks$close_ws(1)
      call gks$close_gks()
      if((terminal.eq.'lanpa').or.(terminal.eq.'LANPA')) then
          call lanpar_text
      endif

      return
      end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
C     TIMEPLOT - Pedro J. Pizarro, 1/16/89
      C
C                                                                    C
C         Uses GKS to plot on a VT240 or a Lanpar Vision II 3222.    C
C                                                                    C
```

```
C    Revisions - Pedro J. Pizarro, 9/1/89
     C
C                                                                       C
C         Now adjusts ordery for borders larger than 1.e7.              C
C                                                                       C
C    Revisions - Pedro J. Pizarro, 6/13/90
     C
C                                                                       C
C         Now accepts the terminal label 't4010' for PC emulation       C
C         of Tek4010 terminals.                                         C
C                                                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC


      subroutine timeplot(narrmx,ntplot,answer,npts,xplot,yplot,
     1    xmin1,xmax1,xmin2,xmax2,terminal)

      real*4 xbordr,ybordr,xplot(narrmx),yplot(narrmx),xpt(4),
     1    ypt(4),zero(4),x(20000),y(20000),xmin1,xmax1,ymin,ymax,
     2    xm(3),ym(3),deltat,orderx,ordery,xmin2,xmax2,
     3    xm2(2)
      integer npts,nunits,nrmndr,nplot
      character*4 answer
      character*5 terminal
      character*10 cxpt(4),cypt(3)

      zero(1)=0.
      zero(2)=0.
      zero(3)=0.
      zero(4)=0.

     write(6,120)
  120 format(5x,'Enter vertical minimum:')
      read(5,*)ymin
      write(6,130)
  130 format(5x,'Enter vertical maximum:')
      read(5,*)ymax

C
C    Since GKS does not like window borders smaller than ~1.e-4,
C    or larger than ~1.e8, and since the scales involved here are
C    smaller than that, adjust ymax and ymin  appropriately
C
      ordery=1.
  320 if((abs(ymax).lt.1.e-4 .and. ymax.ne.0.) .or.
     1    (abs(ymin).lt.1.e-4 .and. ymin.ne.0.)) goto 330

      goto 350

  330 ordery=ordery*10.
      ymax=ymax*10.
      ymin=ymin*10.
      goto 320

  350 if((abs(ymax).gt.1.e7).or.(abs(ymin).gt.1.e7)) goto 360
      goto 400

  360 ordery=ordery/10.
      ymax=ymax/10.
      ymin=ymin/10.
      goto 350

  400 xmin=float(1)
      xmax=float(ntplot)

      do 10 i=1,ntplot
          yplot(i)=yplot(i)*ordery
   10 continue
C
C    Set up borders
C
      xbordr=(xmax-xmin)*7.5/85.
      ybordr=(ymax-ymin)*7.5/85.

      xpt(1)=xmin
      xpt(2)=xmax
      xpt(3)=float(ntplot-npts/2+1)
      ypt(1)=ymin
      ypt(2)=ymax
      ypt(3)=ymin + (ymax-ymin)/2
      xm(1)=1.0
      xm(2)=1.0
      xm(3)=xmin
      ym(1)=ymin
      ym(2)=ymin
      ym(3)=ymin
      xm2(1)=float(ntplot-npts/2+1)
      xm2(2)=xm2(1)
C
C
C    Create axis labels
C
      open(unit=11,file='trajscr.tch',status='new')
      write(11,42)xmin1,ypt(1)/ordery
      write(11,42)xmax1,ypt(2)/ordery
      write(11,42)xmin2,ypt(3)/ordery
      write(11,42)xmax2,ypt(3)/ordery

   42 format(1x,e10.2,1x,e10.2)
      close(unit=11,status='keep')

      open(unit=12,file='trajscr.tch',status='old')
      do 44 i=1,3
          read(12,46)cxpt(i),cypt(i)
   44 continue
      read(12,46)cxpt(4),cypt(3)
   46 format(1x,a10,1x,a10)
      close(unit=12,status='delete')

C
C    Initialize GKS
```

```
C
      if((terminal.eq.'lanpa').or.(terminal.eq.'LANPA')) then
          call lanpar_graph
          call gks$open_gks('sys$error:')
          call gks$open_ws(1,'tta1:',72)
      endif
      if(terminal.eq.'vt240') then
          call gks$open_gks('sys$error:')
          call gks$open_ws(1,0,0)
      endif
      if(terminal.eq.'t4010') then
          call gks$open_gks('sys$error:')
          call gks$open_ws(1,'sys$output:',72)
      endif
      call gks$activate_ws(1)
C
C     Set graphics window size
C
      call gks$set_window(1,xmin-xbordr,xmax+xbordr,
     1    ymin-ybordr,ymax+ybordr)
      call gks$select_xform(1)
C
C     Draw axes
C
      call gks$polyline(2,xpt,ym)
      call gks$polyline(2,xm,ypt)
      call gks$polyline(2,xm2,ypt)
      if(ymax.ge.0. .and. ymin.le.0) call gks$polyline(2,xpt,zero)
C
C     Draw ticks, write labels
C
      call gks$set_pmark_size(1.0)
      call gks$set_pmark_type(2)
      call gks$polymarker(3,xpt,ym)
      call gks$set_text_height(0.015*(2*ybordr+ymax-ymin))
      call gks$set_text_align(1,1)
      call gks$text(1.0,ymin,cxpt(1))
      call gks$text(float(ntplot-npts/2+1),ymin,cxpt(3))
      call gks$set_text_align(3,1)
      call gks$text(float(npts/2),ymin,cxpt(2))
      call gks$text(float(ntplot),ymin,cxpt(4))
      call gks$polymarker(3,xm,ypt)
      call gks$polymarker(3,xm2,ypt)
      call gks$set_text_align(2,5)
      call gks$text(1.0,ypt(1),cypt(1))
      call gks$text(1.0,ypt(2),cypt(2))
      call gks$text(1.0,ypt(3),cypt(3))
      call gks$set_pmark_type(1)

      nplot=ntplot
      nunits = ifix(float(nplot)/20000.)
      nrmndr = mod(nplot,20000)

      if(answer.eq.'line' .or. answer.eq.'LINE') goto 15

      if(nunits.eq.0)goto 54

      do 52 j=1,nunits
          do 50 i=1,20000
              index=(j-1)*20000 + i
              x(i)=xplot(index)
              y(i)=yplot(index)
 50       continue
          call gks$polymarker(20000,x,y)
 52       continue

          if(nrmndr.eq.0)goto 19

 54   do 56 i=1,nrmndr
          index=nunits*20000 + i
          x(i)=xplot(index)
          y(i)=yplot(index)
 56       continue
          call gks$polymarker(nrmndr,x,y)
      goto 19

 15   if(nunits.eq.0)goto 64

      do 62 j=1,nunits
          do 60 i=1,20000
              index=(j-1)*20000 + i
              x(i)=xplot(index)
              y(i)=yplot(index)
 60       continue
          call gks$polyline(20000,x,y)
 62   continue

          if(nrmndr.eq.0)goto 19
 64   do 66 i=1,nrmndr
          index=nunits*20000 + i
          x(i)=xplot(index)
          y(i)=yplot(index)
 66       continue
          call gks$polyline(nrmndr,x,y)

 19   write(6,20)
 20   format(5x,'Press <RETURN> to close this plot:')
      read(5,25)
 25   format(1x)
      call gks$deactivate_ws(1)
      call gks$close_ws(1)
      call gks$close_gks()
      if((terminal.eq.'lanpa').or.(terminal.eq.'LANPA')) then
          call lanpar_text
      endif

C
C     Restore original values of data set
C
      do 1010 i=1,ntplot
          yplot(i)=yplot(i)/ordery
 1010     continue
```

```
      return
      end


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                               C
C    FOUR1 - from Numerical Recipes, 1986, pp. 394-5                            C
C        Replaces DATA by its discrete Fourier transform, if                    C
C        ISIGN is input as 1; or replaces DATA by NN times its                  C
C        inverse discrete Fourier transform, if ISIGN is input                  C
C        as -1.  DATA is a complex array of length NN or,                       C
C        equivalently, a real array of length 2*NN.  NN MUST be                 C
C        an integer power of 2 (this is not checked for!).                      C
C                                                                               C
C    Pedro J. Pizarro, 1/23/89 - made everything real*8                         C
C                                                                               C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine four1(data,nn,isign)

      real*8 wr,wi,wpr,wpi,wtemp,theta
      real*8 data(2*nn),tempr,tempi
      n=2*nn
      j=1
      do 20 i=1,n,2
          if(j.gt.i)then
              tempr=data(j)
              tempi=data(j+1)
              data(j)=data(i)
              data(j+1)=data(i+1)
              data(i)=tempr
              data(i+1)=tempi
          endif
          m=n/2
 10       if((m.ge.2).and.(j.gt.m))then
              j=j-m
              m=m/2
          goto 10
          endif
          j=j+m
 20       continue
      mmax=2
 30   if(n.gt.mmax)then
          istep=2*mmax
          theta=6.28318530717959d0/(isign*mmax)
          wpr=-2.d0*dsin(0.5d0*theta)**2
          wpi=dsin(theta)
          wr=1.d0
          wi=0.d0
          do 50 m=1,mmax,2
              do 40 i=m,n,istep
                  j=i+mmax
                  tempr=wr*data(j)-wi*data(j+1)
                  tempi=wr*data(j+1)+wi*data(j)
                  data(j)=data(i)-tempr
                  data(j+1)=data(i+1)-tempi
                  data(i)=data(i)+tempr
                  data(i+1)=data(i+1)+tempi
 40           continue
          wtemp=wr
          wr=wr*wpr-wi*wpi+wr
          wi=wi*wpr+wtemp*wpi+wi
 50       continue
          mmax=istep
          goto 30
      endif
      return
      end



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                               C
C    LANPAR_GRAPH - Russ Bowers                                                 C
C        Gets Lanpar Vision II 3222 terminal into graphics mode.                C
C                                                                               C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE LANPAR_GRAPH

      character esc,gs
      data esc /27/
      data gs  /29/

         open(unit=9,file='$terminal1',status='old',err=4901)
         write(9,5334),esc,gs
 5334    format(x,a,'1',a)
            close(9)
 4901 continue
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                               C
C    LANPAR_TEXT - Russ Bowers                                                  C
C        Gets Lanpar Vision II 3222 terminal into text mode.                    C
C                                                                               C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE LANPAR_TEXT
      character esc
      data esc /27/

         open(unit=9,file='$terminal1',status='old',err=4900)
         write(9,5333),esc
 5333    format(x,a,'2')
            close(9)
 4900 continue
      END
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                               C
C   PEAK2 - Pedro J. Pizarro, 2/13/89
    C
C                                                                               C
C       Finds maxima and minima.                                                C
C                                                                               C
C   Revisions - Pedro J. Pizarro, 7/12/89
    C
C                                                                               C
C       Added search for left/right maxima/minima.                              C
C       Added option to save peak file.
    C
C                                                                               C
C   Revisions - Pedro J. Pizarro, 9/7/89
    C
C                                                                               C
C       Added output of first and last points.
    C
C                                                                               C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

    subroutine peak2(narrmx,narray,npts,time,intens,tmin1,tmax1,
    1   tmin2,tmax2)

    real*4 intens(narrmx),time(narrmx),tim,tmin1,tmin2,tmax1,tmax2
    integer narrmx,npts
    character*1 pkansr
    character*60 pkfile

    if(intens(1).gt.intens(2)) then
        write(6,100) tmin1,intens(1)
    endif

    if(intens(1).lt.intens(2)) then
        write(6,110) tmin1,intens(1)
    endif

    if(intens(1).eq.intens(2)) then
        write(6,103) tmin1,intens(1)
    endif

    do 50 i=2,narray
        tim = tmin1 + (time(i)-1.)*(tmax1-tmin1)/float(narray-1)
        if((intens(i).gt.intens(i-1)).and.(intens(i).gt.intens(i+1)))
    1       then
            write(6,100) tim,intens(i)

        endif

        if((intens(i).gt.intens(i-1)).and.(intens(i).eq.intens(i+1)))
    1       then
            write(6,101) tim,intens(i)

        endif

        if((intens(i).eq.intens(i-1)).and.(intens(i).gt.intens(i+1)))
    1       then
            write(6,102) tim,intens(i)

        endif

        if((intens(i).lt.intens(i-1)).and.(intens(i).lt.intens(i+1)))
    1       then
            write(6,110) tim,intens(i)

        endif

        if((intens(i).lt.intens(i-1)).and.(intens(i).eq.intens(i+1)))
    1       then
            write(6,111) tim,intens(i)

        endif

        if((intens(i).eq.intens(i-1)).and.(intens(i).lt.intens(i+1)))
    1       then
            write(6,112) tim,intens(i)

        endif
50  continue

    do 60 i=npts-narray+1,npts-1
        tim = tmin2 + (time(i)-float(npts-narray+1))*
    1           (tmax2-tmin2)/float(narray-1)
        if((intens(i).gt.intens(i-1)).and.(intens(i).gt.intens(i+1)))
    1       then
            write(6,100) tim,intens(i)

        endif

        if((intens(i).gt.intens(i-1)).and.(intens(i).eq.intens(i+1)))
    1       then
            write(6,101) tim,intens(i)

        endif

        if((intens(i).eq.intens(i-1)).and.(intens(i).gt.intens(i+1)))
    1       then
            write(6,102) tim,intens(i)

        endif

        if((intens(i).lt.intens(i-1)).and.(intens(i).lt.intens(i+1)))
    1       then
            write(6,110) tim,intens(i)

        endif
```

```
        if((intens(i).lt.intens(i-1)).and.(intens(i).eq.intens(i+1)))
     1       then
             write(6,111) tim,intens(i)

        endif

        if((intens(i).eq.intens(i-1)).and.(intens(i).lt.intens(i+1)))
     1       then
             write(6,112) tim,intens(i)

        endif
 60  continue

     if(intens(npts).gt.intens(npts-1)) then
         write(6,100) tmax2,intens(n)
     endif

     if(intens(npts).lt.intens(npts-1)) then
         write(6,110) tmax2,intens(npts)
     endif

     if(intens(npts).eq.intens(npts-1)) then
         write(6,104) tmax2,intens(npts)
     endif

100  format(5x,'Maximum at ',e15.8,' sec,  ',e15.8,' cm')
101  format(5x,'Left max.  ',e15.8,' sec,  ',e15.8,' cm')
102  format(5x,'Right max. ',e15.8,' sec,  ',e15.8,' cm')
103  format(5x,'First point',e15.8,' sec,  ',e15.8,' cm')
104  format(5x,'Last point ',e15.8,' sec,  ',e15.8,' cm')
110  format(5x,'Minimum at ',e15.8,' sec,  ',e15.8,' cm')
111  format(5x,'Left min.  ',e15.8,' sec,  ',e15.8,' cm')
112  format(5x,'Right min. ',e15.8,' sec,  ',e15.8,' cm')


200  write(6,*) '      Do you want to save this peak record?  Y/N'
     read(5,210) pkansr
210  format(a1)
     if(pkansr.eq.'y' .or. pkansr.eq.'Y') goto 300
     if(pkansr.eq.'n' .or. pkansr.eq.'N') goto 400
     goto 200

300  write(6,*) '      Enter name of peak record file:'
     read(5,310) pkfile
310  format(a60)
     open(unit=11,file=pkfile,status='new')

     write(11,*)pkfile
     write(11,*)

     if(intens(1).gt.intens(2)) then
         write(11,100) tmin1,intens(1)
     endif

     if(intens(1).lt.intens(2)) then
         write(11,110) tmin1,intens(1)
     endif

     if(intens(1).eq.intens(2)) then
         write(11,103) tmin1,intens(1)
     endif

     do 350 i=2,narray
         tim = tmin1 + (time(i)-1.)*(tmax1-tmin1)/float(narray-1)
         if((intens(i).gt.intens(i-1)).and.(intens(i).gt.intens(i+1)))
     1       then
             write(11,100) tim,intens(i)

         endif

         if((intens(i).gt.intens(i-1)).and.(intens(i).eq.intens(i+1)))
     1       then
             write(11,101) tim,intens(i)

         endif

         if((intens(i).eq.intens(i-1)).and.(intens(i).gt.intens(i+1)))
     1       then
             write(11,102) tim,intens(i)

         endif

         if((intens(i).lt.intens(i-1)).and.(intens(i).lt.intens(i+1)))
     1       then
             write(11,110) tim,intens(i)

         endif

         if((intens(i).lt.intens(i-1)).and.(intens(i).eq.intens(i+1)))
     1       then
             write(11,111) tim,intens(i)

         endif

         if((intens(i).eq.intens(i-1)).and.(intens(i).lt.intens(i+1)))
     1       then
             write(11,112) tim,intens(i)

         endif
350  continue

     do 360 i=npts-narray+1,npts-1
         tim = tmin2 + (time(i)-float(npts-narray+1))*
     1           (tmax2-tmin2)/float(narray-1)
         if((intens(i).gt.intens(i-1)).and.(intens(i).gt.intens(i+1)))
     1       then
             write(11,100) tim,intens(i)

         endif

         if((intens(i).gt.intens(i-1)).and.(intens(i).eq.intens(i+1)))
     1       then
```

```
          write(11,101) tim,intens(i)

      endif

      if((intens(i).eq.intens(i-1)).and.(intens(i).gt.intens(i+1)))
     1     then
          write(11,102) tim,intens(i)

      endif

      if((intens(i).lt.intens(i-1)).and.(intens(i).lt.intens(i+1)))
     1     then
          write(11,110) tim,intens(i)

      endif

      if((intens(i).lt.intens(i-1)).and.(intens(i).eq.intens(i+1)))
     1     then
          write(11,111) tim,intens(i)

      endif

      if((intens(i).eq.intens(i-1)).and.(intens(i).lt.intens(i+1)))
     1     then
          write(11,112) tim,intens(i)

      endif
 360  continue

      if(intens(npts).gt.intens(npts-1)) then
          write(11,100) tmax2,intens(n)
      endif

      if(intens(npts).lt.intens(npts-1)) then
          write(11,110) tmax2,intens(npts)
      endif

      if(intens(npts).eq.intens(npts-1)) then
          write(11,104) tmax2,intens(npts)
      endif

      close(unit=11)

 400  return
      end


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     PEAK - Pedro J. Pizarro, 2/9/89
      C
C                                                                      C
C          Finds maxima and minima in FFT, subject to THRESHold.       C
C                                                                      C
C     Revisions - Pedro J. Pizarro, 7/12/89
      C
C                                                                      C
C          Added search for left/right maxima/minima.                  C
C                                                                      C
C     Revisions - Pedro J. Pizarro, 9/7/89
      C
C                                                                      C
C          Added output of first and last points.
      C
C                                                                      C
C     Revisions - Pedro J. Pizarro, 9/14/89
      C
C                                                                      C
C          Made arrays narrmx long, not narrmx/2+1.                    C
C          Now loop over nptsft values, not nptsft/2+1.                C
C                                                                      C
C     Revisions - Pedro J. Pizarro, 6/3/91
      C
C                                                                      C
C          Now output FTMAX.                                           C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine peakf(narrmx,nptsft,fnyqst,freq,freqcy,intens,ftmax)

      real*8 intens(narrmx),freqcy(narrmx),fnyqst,freq,res,
      1   ftmax,filres,thrsh1,thresh
      integer nptsft
      character*1 pkansr
      character*60 pkfile

      write(6,*)'   Enter threshold: '
      read(5,30) thrsh1
      thresh = thrsh1*ftmax
   30 format(d20.10)
      write(6,*)'    Maximum intensity (1.00) = ',ftmax
      write(6,*)'    Number of time domain points transformed = ',nptsft
      write(6,*)'    Nyquist frequency = ',fnyqst,' Hz'
      res = fnyqst*2.0/dfloat(nptsft)
      write(6,*)'    Trajectory Resolution = ',res,' Hz/point'
      write(6,*)'    Reference frequency = ',freq,' Hz'
      write(6,*)'    Threshold = ',thrsh1


      if(intens(1).gt.intens(2)) then
          write(6,100) freqcy(1),intens(1)/ftmax
      endif

      if(intens(1).lt.intens(2)) then
          write(6,110) freqcy(1),intens(1)/ftmax
      endif

      if(intens(1).eq.intens(2)) then
          write(6,103) freqcy(1),intens(1)/ftmax
      endif
```

```
      do 50 i=2,nptsft-1

          if((intens(i).gt.intens(i-1)).and.(intens(i).gt.intens(i+1))
     1        .and. ((dabs(intens(i)-intens(i-1)).ge.thresh)
     2        .or. (dabs(intens(i)-intens(i+1)).ge.thresh))) then
              write(6,100) freqcy(i),intens(i)/ftmax

          endif

          if((intens(i).gt.intens(i-1)).and.(intens(i).eq.intens(i+1))
     1        .and. ((dabs(intens(i)-intens(i-1)).ge.thresh)
     2        .or. (dabs(intens(i)-intens(i+1)).ge.thresh))) then
              write(6,101) freqcy(i),intens(i)/ftmax

          endif

          if((intens(i).eq.intens(i-1)).and.(intens(i).gt.intens(i+1))
     1        .and. ((dabs(intens(i)-intens(i-1)).ge.thresh)
     2        .or. (dabs(intens(i)-intens(i+1)).ge.thresh))) then
              write(6,102) freqcy(i),intens(i)/ftmax

          endif

          if((intens(i).lt.intens(i-1)).and.(intens(i).lt.intens(i+1))
     1        .and. ((dabs(intens(i)-intens(i-1)).ge.thresh)
     2        .or. (dabs(intens(i)-intens(i+1)).ge.thresh))) then
              write(6,110) freqcy(i),intens(i)/ftmax

          endif

          if((intens(i).lt.intens(i-1)).and.(intens(i).eq.intens(i+1))
     1        .and. ((dabs(intens(i)-intens(i-1)).ge.thresh)
     2        .or. (dabs(intens(i)-intens(i+1)).ge.thresh))) then
              write(6,111) freqcy(i),intens(i)/ftmax

          endif

          if((intens(i).eq.intens(i-1)).and.(intens(i).lt.intens(i+1))
     1        .and. ((dabs(intens(i)-intens(i-1)).ge.thresh)
     2        .or. (dabs(intens(i)-intens(i+1)).ge.thresh))) then
              write(6,112) freqcy(i),intens(i)/ftmax

          endif
     50 continue

      if(intens(nptsft).gt.intens(nptsft-1)) then
          write(6,100) freqcy(nptsft),intens(nptsft)/ftmax
      endif

      if(intens(nptsft).lt.intens(nptsft-1)) then
          write(6,110) freqcy(nptsft),intens(nptsft)/ftmax
      endif

      if(intens(nptsft).eq.intens(nptsft-1)) then
          write(6,104) freqcy(nptsft),intens(nptsft)/ftmax
      endif

     100 format(5x,'Maximum at ',f16.0,' Hz, Intensity = ',d10.3)
     101 format(5x,'Left max.  ',f16.0,' Hz, Intensity = ',d10.3)
     102 format(5x,'Right max. ',f16.0,' Hz, Intensity = ',d10.3)
     103 format(5x,'First point',f16.0,' Hz, Intensity = ',d10.3)
     104 format(5x,'Last point ',f16.0,' Hz, Intensity = ',d10.3)
     110 format(5x,'Minimum at ',f16.0,' Hz, Intensity = ',d10.3)
     111 format(5x,'Left min.  ',f16.0,' Hz, Intensity = ',d10.3)
     112 format(5x,'Right min. ',f16.0,' Hz, Intensity = ',d10.3)

     200 write(6,*) '      Do you want to save this peak record?  Y/N'
         read(5,210) pkansr
     210 format(a1)
         if(pkansr.eq.'y' .or. pkansr.eq.'Y') goto 300
         if(pkansr.eq.'n' .or. pkansr.eq.'N') goto 400
         goto 200

     300 write(6,*) '      Enter name of peak record file:'
         read(5,310) pkfile
     310 format(a60)
         open(unit=11,file=pkfile,status='new')

         write(11,*)pkfile
         write(11,*)'   Maximum intensity (1.00) = ',ftmax
         write(11,*)'   # points = ',nptsft,'; Threshold = ',thrsh1
         write(11,*)'   Nyquist frequency = ',fnyqst,' Hz'
         write(11,*)'   Resolution = ',res,' Hz/point'
         write(11,*)'   Reference frequency = ',freq,' Hz'

         if(intens(1).gt.intens(2)) then
             write(11,100) freqcy(1),intens(1)/ftmax
         endif


         if(intens(1).lt.intens(2)) then
             write(11,110) freqcy(1),intens(1)/ftmax
         endif

         if(intens(1).eq.intens(2)) then
             write(11,103) freqcy(1),intens(1)/ftmax
         endif

         do 350 i=2,nptsft-1
             if((intens(i).gt.intens(i-1)).and.(intens(i).gt.intens(i+1))
     1           .and. ((dabs(intens(i)-intens(i-1))).ge.thresh
     2           .or. (dabs(intens(i)-intens(i+1)).ge.thresh))) then
                 write(11,100) freqcy(i),intens(i)/ftmax

             endif

             if((intens(i).gt.intens(i-1)).and.(intens(i).eq.intens(i+1))
     1           .and. ((dabs(intens(i)-intens(i-1))).ge.thresh
     2           .or. (dabs(intens(i)-intens(i+1)).ge.thresh))) then
                 write(11,101) freqcy(i),intens(i)/ftmax
```

```
      endif

      if((intens(i).eq.intens(i-1)).and.(intens(i).gt.intens(i+1))
     1     .and. ((dabs(intens(i)-intens(i-1))).ge.thresh
     2     .or. (dabs(intens(i)-intens(i+1)).ge.thresh))) then
           write(11,102) freqcy(i),intens(i)/ftmax

      endif

      if((intens(i).lt.intens(i-1)).and.(intens(i).lt.intens(i+1))
     1     .and. ((dabs(intens(i)-intens(i-1)).ge.thresh)
     2     .or. (dabs(intens(i)-intens(i+1)).ge.thresh))) then
           write(11,110) freqcy(i),intens(i)/ftmax

      endif

      if((intens(i).lt.intens(i-1)).and.(intens(i).eq.intens(i+1))
     1     .and. ((dabs(intens(i)-intens(i-1)).ge.thresh)
     2     .or. (dabs(intens(i)-intens(i+1)).ge.thresh))) then
           write(11,111) freqcy(i),intens(i)/ftmax

      endif

      if((intens(i).eq.intens(i-1)).and.(intens(i).lt.intens(i+1))
     1     .and. ((dabs(intens(i)-intens(i-1)).ge.thresh)
     2     .or. (dabs(intens(i)-intens(i+1)).ge.thresh))) then
           write(11,112) freqcy(i),intens(i)/ftmax

      endif
 350 continue

      if(intens(nptsft).gt.intens(nptsft-1)) then
          write(11,100) freqcy(nptsft),intens(nptsft)/ftmax
      endif

      if(intens(nptsft).lt.intens(nptsft-1)) then
          write(11,110) freqcy(nptsft),intens(nptsft)/ftmax
      endif

      if(intens(nptsft).eq.intens(nptsft-1)) then
          write(11,104) freqcy(nptsft),intens(nptsft)/ftmax
      endif

      close(unit=11)

 400 return
      end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C     PLOT - Pedro J. Pizarro, 1/18/89
C     C
C          Uses GKS to plot on a VT240 or a Lanpar Vision II 3222.    C
C                                                                     C
C     Revisions - Pedro J. Pizarro, 9/1/89
C     C
C                                                                     C
C          Now adjusts order for borders larger than 1.e7.            C
C                                                                     C
C     Revisions - Pedro J. Pizarro, 9/14/89
C     C
C                                                                     C
C          Now reads xmax, xmin, ymax, and ymin as part of its        C
C          call.  Adjusts x,y orders separately.
C     C
C                                                                     C
C     Revisions - Pedro J. Pizarro, 5/19/90
C     C
C                                                                     C
C          Now leaves original max and min (xmax1,xmin1,ymax1,        C
C          ymin1) intact, and equates subroutine's xmax,xmin,ymax,    C
C          ymin to these, respectively.
C     C
C                                                                     C
C     Revisions - Pedro J. Pizarro, 6/13/90
C     C
C                                                                     C
C          Now accepts the terminal label 't4010' for PC emulation    C
C          of Tek4010 terminals.                                      C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine plotf(narrmx,answer,npts,xplot,yplot,terminal,xmax1,
     1     xmin1,ymax1,ymin1)

      real*4 xbordr,ybordr,xplot(narrmx),yplot(narrmx),xpt(4),
     1     ypt(4),zero(4),x(20000),y(20000),xmin,xmax,ymin,ymax,
     2     xm(3),ym(3),scale,xori,yori,orderx,xmin1,xmax1,ymin1,ymax1
      integer npts,nunits,nrmndr
      character*4 answer
      character*5 terminal
      character*10 cxpt(3),cypt(3)

      zero(1)=0.
      zero(2)=0.
      zero(3)=0.
      zero(4)=0.

      xmax = xmax1
      xmin = xmin1
      ymax = ymax1
      ymin = ymin1

C     write(6,100)
C100 format(5x,'Enter window scale:')
C     read(5,*)scale
C     write(6,110)
C110 format(5x,'Enter horizontal center of plot:')
C     read(5,*)xori
C     write(6,120)
```

```
C120 format(5x,'Enter vertical center of plot:')
C    read(5,*)yori
C
C    xmax=xori+scale
C    xmin=xori-scale
C    ymax=yori+scale
C    ymin=yori-scale

C
C    Since GKS does not like window borders smaller than ~1.e-4,
C    or larger that ~1.e8, and since the scales involved here are
C    smaller than that, adjust xmax, xmin, ymax, and ymin appropriately
C


     orderx=1.
     ordery=1.
 300 if((abs(xmax).lt.1.e-4 .and. xmax.ne.0.) .or.
     1    (abs(xmin).lt.1.e-4 .and. xmin.ne.0.)) goto 310

     goto 350

 310 orderx=orderx*10.
     xmax=xmax*10.
     xmin=xmin*10.
     goto 300

 350 if((abs(xmax).gt.1.e7) .or. (abs(xmin).gt.1.e7))
     1    goto 360

     goto 370

 360 orderx=orderx/10.
     xmax=xmax/10.
     xmin=xmin/10.
     goto 350

 370 if((abs(ymax).lt.1.e-4 .and. ymax.ne.0.) .or.
     1    (abs(ymin).lt.1.e-4 .and. ymin.ne.0.)) goto 375

     goto 380

 375 ordery=ordery*10.
     ymax=ymax*10.
     ymin=ymin*10.
     goto 370

 380 if((abs(ymax).gt.1.e7) .or. (abs(ymin).gt.1.e7))
     1    goto 385

     goto 400

 385 ordery=ordery/10.
     ymax=ymax/10.
     ymin=ymin/10.
     goto 380
```

```
C
C    Set up borders
C
 400 xbordr=(xmax-xmin)*7.5/85.
     ybordr=(ymax-ymin)*7.5/85.

     xpt(1)=xmin
     xpt(2)=xmax
     xpt(3)=xmin + (xmax-xmin)/2
     ypt(1)=ymin
     ypt(2)=ymax
     ypt(3)=ymin + (ymax-ymin)/2
     xm(1)=xmin
     xm(2)=xmin
     xm(3)=xmin
     ym(1)=ymin
     ym(2)=ymin
     ym(3)=ymin
C
C
C    Create axis labels
C
     open(unit=11,file='trajscr.tch',status='new')
     do 40 i=1,3
          write(11,42)xpt(i)/orderx,ypt(i)/ordery
  40 continue
  42 format(1x,e10.2,1x,e10.2)
     close(unit=11,status='keep')

     open(unit=12,file='trajscr.tch',status='old')
     do 44 i=1,3
          read(12,46)cxpt(i),cypt(i)
  44 continue
  46 format(1x,a10,1x,a10)
     close(unit=12,status='delete')

C
C    Initialize GKS
C
     if((terminal.eq.'lanpa').or.(terminal.eq.'LANPA')) then
          call lanpar_graph
          call gks$open_gks('sys$error:')
          call gks$open_ws(1,'tta1:',72)
     endif
     if(terminal.eq.'vt240') then
          call gks$open_gks('sys$error:')
          call gks$open_ws(1,0,0)
     endif
     if(terminal.eq.'t4010') then
          call gks$open_gks('sys$error:')
          call gks$open_ws(1,'sys$output:',72)
     endif
     call gks$activate_ws(1)
C
C    Set graphics window size
C
```

```
      call gks$set_window(1,xmin-xbordr,xmax+xbordr,
     1    ymin-ybordr,ymax+ybordr)
      call gks$select_xform(1)
C
C     Draw axes
C
      call gks$polyline(2,xpt,ym)
      call gks$polyline(2,xm,ypt)
      if(xmax.ge.0. .and. xmin.le.0) call gks$polyline(2,zero,ypt)
      if(ymax.ge.0. .and. ymin.le.0) call gks$polyline(2,xpt,zero)
C
C     Draw ticks, write labels
C
      call gks$set_pmark_size(1.0)
      call gks$set_pmark_type(2)
      call gks$polymarker(3,xpt,ym)
      call gks$set_text_height(0.015*(2*ybordr+ymax-ymin))
      call gks$set_text_align(2,1)
      call gks$text(xpt(1),ymin,cxpt(1))
      call gks$text(xpt(2),ymin,cxpt(2))
      call gks$text(xpt(3),ymin,cxpt(3))
      call gks$polymarker(3,xm,ypt)
      call gks$set_text_align(2,5)
      call gks$text(xmin,ypt(1),cypt(1))
      call gks$text(xmin,ypt(2),cypt(2))
      call gks$text(xmin,ypt(3),cypt(3))
      call gks$set_pmark_type(1)

      nunits = ifix(float(npts)/20000.)
      nrmndr = mod(npts,20000)

      if(answer.eq.'line' .or. answer.eq.'LINE') goto 15

      if(nunits.eq.0) goto 54

      do 52 j=1,nunits
          do 50 i=1,20000
              index=(j-1)*20000 + i
              x(i)=xplot(index)*orderx
              y(i)=yplot(index)*ordery
 50       continue
          call gks$polymarker(20000,x,y)
 52       continue

          if(nrmndr.eq.0)goto 19

 54   do 56 i=1,nrmndr
          index=nunits*20000+i
          x(i)=xplot(index)*orderx
          y(i)=yplot(index)*ordery
 56       continue
          call gks$polymarker(nrmndr,x,y)
      goto 19

 15   if(nunits.eq.0) goto 64

      do 62 j=1,nunits
          do 60 i=1,20000
              index=(j-1)*20000 + i
              x(i)=xplot(index)*orderx
              y(i)=yplot(index)*ordery
 60       continue
          call gks$polyline(20000,x,y)
 62   continue

          if(nrmndr.eq.0)goto 19
 64   do 66 i=1,nrmndr
          index=nunits*20000+i
          x(i)=xplot(index)*orderx
          y(i)=yplot(index)*ordery
 66       continue
          call gks$polyline(nrmndr,x,y)

 19   write(6,20)
 20   format(5x,'Press <RETURN> to close this plot:')
      read(5,25)
 25   format(1x)
      call gks$deactivate_ws(1)
      call gks$close_ws(1)
      call gks$close_gks()
      if((terminal.eq.'lanpa').or.(terminal.eq.'LANPA')) then
          call lanpar_text
      endif

      return
      end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     FFT_COMPLEX - Pedro J. Pizarro, 9/14/89                          C
C                                                                      C
C         Given a complex time-domain record with real part TREAL      C
C         and imaginary part TIMAG, each with NPTSFT points,           C
C         outputs the complex FFT as its real part FTREAL and its      C
C         imaginary part FTIMAG.  The sense of the FFT is set by       C
C         ISIGN, since this is the sign in the FFT exponential.        C
C         FTREAL and FTIMAG each have NPTSFT points; point 1           C
C         corresponds to -fnyqst+res, point NPTSFT/2 corresponds       C
C         to zero frequency, and point nptsft corresponds to           C
C         fnyqst.  Remember that FT(-fnyqst) = FT(fnyqst).             C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine fft_complex(treal,timag,nptsft,ftreal,ftimag,isign)

      parameter(nmax=132000)

      real*8 data(2*nmax),treal(nptsft),timag(nptsft),ftreal(nptsft),
     1    ftimag(nptsft)

      integer nptsft,isign
```

```
      do 10 i=1,nptsft
           data(2*i-1) = treal(i)
           data(2*i) = timag(i)
 10   continue

      call four1(data,nptsft,isign)

      do 20 i=1,-1+nptsft/2
           ftreal(i) = data(nptsft+2*i+1)
           ftimag(i) = data(nptsft+2*i+2)
 20   continue

      do 30 i=nptsft/2,nptsft
           ftreal(i) = data(2*i-nptsft+1)
           ftimag(i) = data(2*i-nptsft+2)
 30   continue

      if(isign.eq.-1) then
          do 40 i=1,nptsft
               ftreal(i) = ftreal(i)/nptsft
               ftimag(i) = ftimag(i)/nptsft
 40       continue
      endif

      return
      end
```

A.4 IRICE01.FOR, written for DOS/Windows systems

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C   TRAJ - Pedro J. Pizarro, 1/26/89
    C
C                                  C
C       Given a desired frequency resolution RES, a number of         C
C       Fourier spectrum points NPTSFT, and a trajectory              C
C       simulation time step DT, along with initial values            C
C       for time, positions and velocities (all in the                C
C       file TRAJ.PAR):      TRAJ  computes classical ion             C
C       trajectories and saves NPTSFT points spaced in time           C
C       so that the Nyquist frequency of their FFT will be            C
C       RES*NPTSFT/2.  Saves these points in FILE2, with              C
C       NPTSFT and FNYQST=RES*NPTSFT/2 in the first line.             C
C       Saves NARRAY-long records of the beginning and end            C
C       of the trajectory in FILE1, with NARRAY in                    C
C       the first line of the file, and tmin1, tmax1,tmin2,           C
C       and tmax2 in the last line.  Both data files are              C
C       unformatted, to be processed and plotted by traj_ft.          C
C                                                                     C
C   PJP - revisions - 4/17/89                                         C
C                                                                     C
C       Added capability of ICR in X or Y direction, by CW            C
C       or pulse.
  C
C                                                                     C
C   PJP - revisions - 7/6/89                                          C
C                                                                     C
C       Added common blocks /ion/ and /rkz4/.
  C
C                                                                     C
C   PJP - revisions - 7/12/89                                         C
C                                                                     C
C       Program now writes new versions of TRAJ.PAR and               C
C       TRAJB2.PAR at the end of the trajectory simulation.           C
C                                                                     C
C   PJP - revisions - 8/24/89                                         C
C                                                                     C
C       No longer writes new version of TRAJ.PAR (commented           C
C       out).  Only t,x,y,z,vx,vy,vz written to TRAJB2.PAR.           C
C                                                                     C
C   PJP - revisions - 10/9/89                                         C
C                                                                     C
C       Now calculates quantum numbers n,k,l for the initial          C
C       and final x,v.  Lines relevant to this change are set         C
C       aside between pairs of C####.                                 C
C                                                                     C
C   PJP - revisions - 10/23/91                                        C
C                                                                     C
C       TRAJTSG.FOR - Includes calculation of magnetization in        C
C       Transverse Stern-Gerlach experiment (TSG).  A set of 6        C
C       coupled differential equations is now solved.  The            C
C       components of the spin magnetic moment are MX=X(4),           C
C       MY=X(5), and MZ=X(6).  GRADB is now the gradient              C
C       strength of the TSG rf magnetic field.  This version          C
C       does not calculate magnetic bottle effects.  The              C
C       initial spin orientation is MX=SPIN*2*MAGMOM, MZ=MY=0,        C
C       and dM/dt=0.  CWFREQ is now used for the frequency of         C
C       the TSG rf field.                                             C
C                                                                     C
C   PJP - revisions - 10/29/91                                        C
C                                                                     C
C       Now reads SPINX, SPINY, and SPINZ, the spin orientation       C
C       vector coordinates, from the end of the TRAJ.PAR file.        C
C       The program uses these orientations, normalizing the          C
C       vector to one (i.e., the vector need not be input as          C
C       normalized).                                                  C
C                                                                     C
C   PJP - revisions - 11/5/91                                         C
C                                                                     C
C       Now has a separate time-domain array length MARRAY for        C
C       the spin trajectory files.  Need MARRAY <= NARRAY; if         C
```

```
C       not, effective MARRAY will be NARRAY.
C
C
C   PJP - revisions - 11/11/91
C
C       Now reduces the number of time-domain points saved by
C       using the integers ISPACE and ISPIN for the space and
C       spin calculations, respectively.  Over NARRAY (or
C       MARRAY) every ISPACE (or ISPIN) point is saved,
C       beginning with the first point.  NSPACE and NSPIN are
C       the counters used, respectively.
C
C
C   PJP - revisions - 11/30/91
C
C       Changed variable CWX to CWPHAS, which is the phase
C       of the TSG rf gradient field.  This is input in degrees
C       and immediately converted to radians.
C
C
C   PJP - revisions - 12/17/91
C
C       Changed DERIVS to calculate dM/dt from M cross B.
C
C
C   PJP - revisions - 12/18/91
C
C       Renormalize the spin vector M at each point.
C
C   PJP - revisions - 12/18/91
C
C       Store dM/dt in DERIVS as v(i),i=4,5,6.
C
C       Corrected renormalization of spin (use spin magnitude
C       SPNMAG).
C
C   PJP - revisions - 12/21/91
C
C       Fixed calculation of QUANTN and QUANTL.  This is now
C       done in the subroutine QUANTS, one of whose arguments
C       is QSIGN, the sign of the charge Q.  Calculate
C       quantum numbers throughout simulation and output their
C       trajectories in FILE1QQ and FILE2QQ.
C
C
C   PJP - revisions - 12/22/91
C
C       Moved renormalization of dM/dt back to the main
C       program, out of DERIVS.
C
C
C   PJP - revisions - 12/23/91
C
C       Removed CWY, PULSEX, and PULSEY from the common block
C       ION.  Added SPNMAG to the ION block.
C
C       Changed the initial phase of the rf gradient field from
C       CWPHAS to CWPHS0, input in degrees and converted
C       to radians.  Changed CWY to PHSHFT, the increment by
C       which the phase is shifted at each phase shifting time.
C       Changed PULSEX to PHSFRQ, the frequency at which the
C       phase is shifted (i.e., the phase is shifted by DELPHS
C       every 1/PHSFRQ seconds).  Changed PULSEY to DLSHFT, a
C       fraction of a period of PHSFRQ by which the sequence of
C       phase shifts may be shifted.  The instantaneous TSG rf
C       phase CWPHAS (which is still in the block ION) is thus:
C         CWPHAS = CWPHS0 + PHSHFT*DINT(DLSHFT+(T-T1)*PHSFRQ).
C       This operation is carried out in the subroutine PSHIFT,
C       which uses the common block RFPHAS.
C       The renormalization of the spin vector and dM/dt is now
C       carried out in the subroutine RENORM.
C
C   PJP - revisions - 12/28/91
C
C       Fixed bug: read in CWPHS0, not CWPHAS, from TRAJ.PAR.
C
C   PJP - revisions - 1/6/92
C
C       Fixed bug: convert CWPHS0 to radians (not CWPHAS).
C
C   PJP - revisions - 1/16/92
C
C       TRAJ2COIL.FOR - rf excitation is provided by two coils
C       oscillating at the rf frequency CWFREQ with phases
C       CWPHS1 and CWPHS3, and amplitude modulated at the
C       frequency AMFREQ with phases CWPHS2 and CWPHS4.  Phase
C       shifting, and hence calls to PSHIFT, have been
C       eliminated; the appropriate variables have been left in
C       for convenience.  AMFREQ, CWPHS2, CWPHS3, and CWPHS4
C       are now read from the end of TRAJ.PAR and have been
C       added to the common block ION.
C       (Changed CWPHAS to CWPHS1.)
C
C   PJP - revisions - 1/21/92
C
C       Corrected magnetic field in DERIVS and RENORM.
C
C   PJP - revisions - 7/13/92
C
C       TRAJ2COIL_NORENORM - Disabled renormalization of M and
C       DM/DT (i.e., disabled RENORM).
C
C      PJP - revisions - 7/13/92
C
C             TRAJ2COILB1 - Include homogeneous B1 field for a normal
C             NMR spin lock.  B1 is input in tesla; its field
C       oscillates with frequency B1FREQ and phase B1PHAS.  Its
C       axis is defined by the angle B1ANGL (counterclockwise
```

```
C        angle about z, with the z axis at zero).                        C
C                                                                        C
C       PJP - revisions - 12/14/92                                       C
C                                                                        C
C              IRICE01 - IBM PC version.  Changed name of common block C
C              /rkz4/ to /rkzvar/.                                       C
C                                                                        C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      program irice01

      implicit none
      integer nmax,nvar
      parameter (nmax=6)
      parameter (nvar=6)
      double precision t,t1,dt,res,fnyqst,v(nmax),dv(nmax),b(3),
     1       x(nmax),q,m,magmom,spin,v0,d,b0,gradb,grade,b2,
     2       cwfreq,cwphs1,phshft,phsfrq,dlshft,pi,gamma,qsign
C####
     3       ,hbar,wz,wc,wm,wp,quantn,quantk,quantl,mass
C####
     4       ,amfreq,cwphs2,cwphs3,cwphs4
     5       ,b1,b1freq,b1phas,b1angl
      double precision spinx,spiny,spinz,spnorm,spfact,spnmag
      double precision oldphs,cwphs0
      real*4 tmin1,tmax1,tmin2,tmax2
      integer ncount,narray,nptsft,nperft,nft,nstep,nnext,marray
      integer indx1,indx2,ispace,ispin,nspace,nspin,i,k
      character*60 file1,file2
      integer*2 ihr,imin,isec,i100th,iyr,imon,iday

C
C     Common blocks (added on 7/6/89)
C
      common/ion/ q,m,magmom,spin,v0,d,b0,b2,gradb,grade,cwfreq,
     1       cwphs1,gamma,spnmag,amfreq,cwphs2,cwphs3,cwphs4
     5       ,b1,b1freq,b1phas,b1angl
      common/rkzvar/ t,x,v,dt
      common/quantc/ wp,wm,wz,hbar,quantn,quantl,quantk,qsign,mass
      common/rfphas/ oldphs,cwphs0,phshft,dlshft,t1,phsfrq,pi

      call gettim(ihr,imin,isec,i100th)
      call getdat(iyr,imon,iday)

      open(unit=15,file='irice.out',status='new',form='formatted')

      write(6,*) 'Beginning execution of program IRICE01...'
      write(15,*) 'Beginning execution of program IRICE01...'
C
C     pi = 3.14159 26535 89793 23846 26433 83279 50288 41971 69399 37511
C
      pi = 3.14159265358979
C####
C
C     hbar = 1.0545887d-34 J sec
C
      hbar = 1.0545887d-34
C####

C
C     Read parameters in traj.par
C            t1, dt in sec
C            res in Hz
C            nptsft a power of 2
C            narray < 47059
C            x(i) in centimeter
C            vstart(i) in centimeter/sec
C
      open(unit=8,file='traj.par',status='old')
      read(8,1) file1
      read(8,1) file2
 1    format(1x,a60)
C
C     Get length of filenames
C
      indx1 = index(file1,' ')
      if(indx1.eq.0) indx1=61
      indx1 = indx1 - 1
      indx2 = index(file2,' ')
      if(indx2.eq.0) indx2=61
      indx2 = indx2 - 1
C
C     Open files
C
      open(unit=9,file=file1//'.dat',status='new',form='unformatted')
      open(unit=10,file=file2//'.dat',status='new',form='unformatted')
      open(unit=11,file=file1(1:indx1)//'mm.dat',status='new',
     1       form='unformatted')
      open(unit=12,file=file2(1:indx2)//'mm.dat',status='new',
     1       form='unformatted')
      open(unit=13,file=file1(1:indx1)//'qq.dat',status='new',
     1       form='unformatted')
      open(unit=14,file=file2(1:indx2)//'qq.dat',status='new',
     1       form='unformatted')

      read(8,6) t1
      read(8,6) dt
      read(8,6) res
      read(8,2) nptsft
      fnyqst = res*dfloat(nptsft)/2.0d+0
 2    format(1x,i10)
      read(8,3) narray
 3    format(1x,i20)
      do 4 i=1,3
            read(8,6) x(i)
 4    continue
      do 5 i=1,3
            read(8,6) v(i)
 5    continue
 6    format(1x,d30.15)
C
C     Set up charge (multiple of e), ionic mass (in kg)
```

```
C            Note:   e = 1.6021892D-19 coulomb
C                    mass of electron = 9.109534D-31 kg
C                    mass of proton = 1.6726485D-27 kg
C                    spin = 1. for up, -1. for down (spin-1/2)
C                    "magmom" = magnetic moment
C                    electron magmom = 9.284832d-17 erg/tesla
C                    proton magmom = 1.4106171d-19 erg/tesla
C                    gyromagnetic ratio = gamma = 2*magmom/hbar
C
      read(8,6) q
      qsign = q/dabs(q)
      read(8,6) m
      mass = m
      read(8,6) spin
      read(8,6) magmom
      gamma = (2.d+0*magmom/hbar)*1.d-07      !units are 1/(sec*T)
C
C     Penning trap: v0 in volt, d in centimeter, b0 in tesla
C
      read(8,6) v0
      read(8,6) d
      read(8,6) b0
C
C     Gradient fields scaled to reach a maximum percentage of the
C            static fields at distance d from the origin.
C            Read ratios for "gradb", "grade"; compute gradients.
C            Units of "gradb" are tesla/centimeter.
C            Units of "grade" are volt/centimeter.
C
      read(8,6) gradb
CCC## gradb = gradb*b0/d
      read(8,6) grade
      grade = grade*v0/d
C
C     Magnetic bottle: b2 in tesla/meter**2 = gauss/cm**2
C
      read(8,6) b2
C
C     Convert b2 to tesla/centimeter**2, as required by DERIVS
C
      b2 = b2*1.d-4
C
C     Parameters for TSG rf:  cwfreq, phsfrq in Hz
C                             cwphs0, phshft in degrees
C                             dlshft is fraction of phsfrq period
C
      read(8,6) cwfreq
      read(8,6) cwphs0
      read(8,6) phshft
      read(8,6) phsfrq
      read(8,6) dlshft
C
C     Convert cwfreq to radians/sec
C
      cwfreq = 2.d+0*pi*cwfreq
C
C     Convert cwphs0, phshft to radians
C
      cwphs0 = cwphs0*pi/180.d+0
      phshft = phshft*pi/180.d+0
C
C     Initialize CWPHS1
C
      cwphs1 = cwphs0
C
C     Read the time-domain array length for the spin files
C
      read(8,3) marray
      read(8,3) ispace
      read(8,3) ispin
C
C     Read the spin orientation vector, and normalize
C
      read(8,6) spinx
      read(8,6) spiny
      read(8,6) spinz
      spnorm = dsqrt(spinx**2 + spiny**2 + spinz**2)
C
C     Read frequency and phases of the rf and amplitude modulation
C            AMFREQ in Hz; CWPHS2, CWPHS3, CWPHS4 in degrees
C
      read(8,6) amfreq
      read(8,6) cwphs2
      read(8,6) cwphs3
      read(8,6) cwphs4
C
C     Convert amfreq to radians/sec; CWPHS2, CWPHS3,
C            CWPHS4 to radians
C
      amfreq = 2.d+0*pi*amfreq
      cwphs2 = cwphs2*pi/180.d+0
      cwphs3 = cwphs3*pi/180.d+0
      cwphs4 = cwphs4*pi/180.d+0
C
C     NMR B1 parameters:
C
      read(8,6) b1                                    !tesla
      read(8,6) b1freq                                !Hz
      read(8,6) b1phas                                !degree
      read(8,6) b1angl                                !degree
C
C     Convert B1FREQ to radians/sec; B1PHAS, B1ANGL to radians
C
      b1freq = 2.d+0*pi*b1freq
      b1phas = b1phas*pi/180.d+0
      b1angl = b1angl*pi/180.d+0
C
C     Close TRAJ.PAR
C
      close(unit=8)
C####
C
```

```
C     Calculate the normal mode frequencies (in the absence of b2)
C            (d/100. is d in m, not cm)
C
      wz = sqrt(q*v0/(m*((d/100.)**2)))
      wc = q*b0/m
      wm = 0.5*wc - 0.5*sqrt(wc*wc - 2*wz*wz)
      wp = wc-wm
C
C     Output frequencies
C
      write(6,*)'vc = ',wc/(2.*pi),' Hz'
      write(6,*)'vp = ',wp/(2.*pi),' Hz'
      write(6,*)'vm = ',wm/(2.*pi),' Hz'
      write(6,*)'vz = ',wz/(2.*pi),' Hz'
      write(6,*)'v0 = ',gamma*b0/(2.*pi),' Hz'
C     pi time: use b1/2 to select one of two rotating components
      write(6,*)'pi time = ',2.*pi/(gamma*b1),' sec'
      write(6,*)
      write(6,*)'rf frequency = ',cwfreq/(2.*pi),' Hz'
      write(6,*)'am frequency = ',amfreq/(2.*pi),' Hz'
      write(6,*)'Coil 1 (y-z) rf phase = ',cwphs1*180./pi,' deg'
      write(6,*)'             am phase = ',cwphs2*180./pi,' deg'
      write(6,*)'Coil 2 (z-x) rf phase = ',cwphs3*180./pi,' deg'
      write(6,*)'             am phase = ',cwphs4*180./pi,' deg'
      write(6,*)'b1 rf frequency = ',b1freq/(2.*pi),' Hz'
      write(6,*)'b1 rf phase = ',b1phas*180./pi,' deg'
      write(6,*)'b1 rf axis angle = ',b1angl*180./pi,' deg'
      write(15,*)'vc = ',wc/(2.*pi),' Hz'
      write(15,*)'vp = ',wp/(2.*pi),' Hz'
      write(15,*)'vm = ',wm/(2.*pi),' Hz'
      write(15,*)'vz = ',wz/(2.*pi),' Hz'
      write(15,*)'v0 = ',gamma*b0/(2.*pi),' Hz'
C     pi time: use b1/2 to select one of two rotating components
      write(15,*)'pi time = ',2.*pi/(gamma*b1),' sec'
      write(15,*)
      write(15,*)'rf frequency = ',cwfreq/(2.*pi),' Hz'
      write(15,*)'am frequency = ',amfreq/(2.*pi),' Hz'
      write(15,*)'Coil 1 (y-z) rf phase = ',cwphs1*180./pi,' deg'
      write(15,*)'             am phase = ',cwphs2*180./pi,' deg'
      write(15,*)'Coil 2 (z-x) rf phase = ',cwphs3*180./pi,' deg'
      write(15,*)'             am phase = ',cwphs4*180./pi,' deg'
      write(15,*)'b1 rf frequency = ',b1freq/(2.*pi),' Hz'
      write(15,*)'b1 rf phase = ',b1phas*180./pi,' deg'
      write(15,*)'b1 rf axis angle = ',b1angl*180./pi,' deg'
C
C     Set up spin
C
      spfact = 2.d+0*spin*magmom
      spnmag = dabs(spfact)
      x(4) = spfact*spinx/spnorm
      x(5) = spfact*spiny/spnorm
      x(6) = spfact*spinz/spnorm
      t = t1
      oldphs = 0.d+0
C     call pshift(cwphs1,t)
      call renorm
C
C     Load starting values
C
      tmin1=sngl(t1)
      t=t1
      write(9) narray
      write(9) sngl(x(1)),sngl(x(2)),sngl(x(3))
      nspace = 1
      write(11) narray
      write(11) sngl(x(4)),sngl(x(5)),sngl(x(6))
      nspin = 1
      ncount=1
      nft=1
      call quants
      write(13) narray
      write(13) sngl(quantn),sngl(quantl),sngl(quantk)
      write(10) nptsft,fnyqst
      write(10) x(1),x(2),x(3),t
      write(12) nptsft,fnyqst
      write(12) x(4),x(5),x(6),t
      write(14) nptsft,fnyqst
      write(14) quantn,quantl,quantk,t
C####
C
C     Output initial position, velocity, and quantum numbers
C            (factor of 1.d-4 in quant(n,k,l) needed since
C            x in cm, v in cm/sec, so result is (cm/m)**2;
C            1.d-4 makes the quantum numbers dimensionless)
C
      write(6,*)'t = ',t,' sec'
      write(6,*)'x 1-3:',x(1),x(2),x(3),' cm'
      write(6,*)'v 1-3:',v(1),v(2),v(3),' cm/sec'
      write(6,*)'n,l,k:',quantn,quantl,quantk
      write(6,*)'mx,my,mz:',x(4),x(5),x(6)
      write(6,*)'vmx,vmy,vmz:',v(4),v(5),v(6)
      write(15,*)'t = ',t,' sec'
      write(15,*)'x 1-3:',x(1),x(2),x(3),' cm'
      write(15,*)'v 1-3:',v(1),v(2),v(3),' cm/sec'
      write(15,*)'n,l,k:',quantn,quantl,quantk
      write(15,*)'mx,my,mz:',x(4),x(5),x(6)
      write(15,*)'vmx,vmy,vmz:',v(4),v(5),v(6)
C####
C
C     Calculate nperft = number trajectory points per FT point
C
      nperft = nint( 1.0 / ( dt*res*dfloat(nptsft) ) )
      nstep = nperft*(nptsft-1)+1
      nnext=nft*nperft + 1
C
C     Take first NARRAY steps, filling tt1,traj1
C
      do 13 k=2,narray
C            call pshift(cwphs1,t)    !Calculate TSG rf phase
             call rkz4
C            call renorm      !Renormalize M, dM/dt
```

```
            ncount=ncount+1
            if(nspace.eq.ispace) then
              write(9) sngl(x(1)),sngl(x(2)),sngl(x(3))
              call quants
              write(13) sngl(quantn),sngl(quantl),sngl(quantk)
              nspace = 1
            else
              nspace = nspace + 1
            endif
            if(k.le.marray) then
                    if(nspin.eq.ispin) then
                      write(11) sngl(x(4)),sngl(x(5)),sngl(x(6))
                      nspin = 1
                    else
                      nspin = nspin + 1
                    endif
            endif
            if(ncount.eq.nnext) then
                    write(10) x(1),x(2),x(3),t
                    write(12) x(4),x(5),x(6),t
                    call quants
                    write(14) quantn,quantl,quantk,t
                    nft=nft+1
                    nnext=nft*nperft+1
            endif
 13   continue
            tmax1=sngl(t)
C
C     Take next NSTEP-2*NARRAY steps
C
      do 113 k=narray+1,nstep-narray
C             call pshift(cwphs1,t)   !Calculate TSG rf phase
              call rkz4
C             call renorm     !Renormalize M, dM/dt

              ncount=ncount+1
              if(ncount.eq.nnext) then
                      write(10) x(1),x(2),x(3),t
                      write(12) x(4),x(5),x(6),t
                      call quants
                      write(14) quantn,quantl,quantk,t
                      nft=nft+1
                      nnext=nft*nperft+1
              endif
 113  continue
C
C     Reset NSPACE and NSPIN
C
      nspace = ispace
      nspin = ispin
C
C     Take final NARRAY steps, filling tt2,traj2
C
      tmin2 = sngl(t+dt)
      do 213 k=1,narray
C             call pshift(cwphs1,t)   !Calculate TSG rf phase
              call rkz4
C             call renorm     !Renormalize M, dM/dt

              ncount=ncount+1
              if(nspace.eq.ispace) then
                write(9) sngl(x(1)),sngl(x(2)),sngl(x(3))
                call quants
                write(13) sngl(quantn),sngl(quantl),sngl(quantk)
                nspace = 1
              else
                nspace = nspace + 1
              endif
              if(narray-marray.lt.k) then
                      if(nspin.eq.ispin) then
                        write(11) sngl(x(4)),sngl(x(5)),sngl(x(6))
                        nspin = 1
                      else
                        nspin = nspin + 1
                      endif
              endif
              if(ncount.eq.nnext) then
                      write(10) x(1),x(2),x(3),t
                      write(12) x(4),x(5),x(6),t
                      call quants
                      write(14) quantn,quantl,quantk,t
                      nft=nft+1
                      nnext=nft*nperft+1
              endif
 213  continue
              tmax2=sngl(t)

      write(9) tmin1,tmax1,tmin2,tmax2
      write(11) tmin1,tmax1,tmin2,tmax2
      write(13) tmin1,tmax1,tmin2,tmax2
C
C     Output x,v,t to screen
C
C     write(6,*) 'x  = ',x(1),' cm'
C     write(6,*) 'y  = ',x(2),' cm'
C     write(6,*) 'z  = ',x(3),' cm'
C     write(6,*) 'vx = ',v(1),' cm/sec'
C     write(6,*) 'vy = ',v(2),' cm/sec'
C     write(6,*) 'vz = ',v(3),' cm/sec'
C     write(6,*) 't  = ',t,' sec'
C     write(15,*) 'x  = ',x(1),' cm'
C     write(15,*) 'y  = ',x(2),' cm'
C     write(15,*) 'z  = ',x(3),' cm'
C     write(15,*) 'vx = ',v(1),' cm/sec'
C     write(15,*) 'vy = ',v(2),' cm/sec'
C     write(15,*) 'vz = ',v(3),' cm/sec'
C     write(15,*) 't  = ',t,' sec'
C####
C
C     Output initial position, velocity, and quantum numbers
C             (factor of 1.d-4 in quant(n,k,l) needed since
C             x in cm, v in cm/sec, so result is (cm/m)**2;
```

```
C            1.d-4 makes the quantum numbers dimensionless)
C
      write(6,*)'t = ',t,' sec'
      write(6,*)'x 1-3:',x(1),x(2),x(3),' cm'
      write(6,*)'v 1-3:',v(1),v(2),v(3),' cm/sec'
      write(15,*)'t = ',t,' sec'
      write(15,*)'x 1-3:',x(1),x(2),x(3),' cm'
      write(15,*)'v 1-3:',v(1),v(2),v(3),' cm/sec'
      call quants
      write(6,*)'n,l,k:',quantn,quantl,quantk
      write(6,*)'mx,my,mz:',x(4),x(5),x(6)
      write(6,*)'vmx,vmy,vmz:',v(4),v(5),v(6)
      write(15,*)'n,l,k:',quantn,quantl,quantk
      write(15,*)'mx,my,mz:',x(4),x(5),x(6)
      write(15,*)'vmx,vmy,vmz:',v(4),v(5),v(6)
C####

      close(unit=9)
      close(unit=10)
      close(unit=11)
      close(unit=12)
      close(unit=13)
      close(unit=14)

      write(6,*)
      write(6,200) imon,iday,iyr,ihr,imin,isec,i100th
      write(15,*)
      write(15,200) imon,iday,iyr,ihr,imin,isec,i100th
      call getdat(iyr,imon,iday)
      call gettim(ihr,imin,isec,i100th)
      write(6,202) imon,iday,iyr,ihr,imin,isec,i100th
      write(15,202) imon,iday,iyr,ihr,imin,isec,i100th
 200  format(1x,'Program started at: ',i2,'-',i2.2,'-',i4,
     1    1x,i2,':',i2.2,':',i2.2,'.',i2.2)
 202  format(1x,'Program ended at:   ',i2,'-',i2.2,'-',i4,
     1    1x,i2,':',i2.2,':',i2.2,'.',i2.2)

      end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C   RKZ4 - Pedro J. Pizarro, 2/17/89
    C
C                                                                      C
C       Based on J. A. Zonneveld, "Automatic Numerical                 C
C       Integration" (Mathematical Centre Tracts No. 8,                C
C       Mathematisch Centrum, Amsterdam, 1964),   pp. 32-40.           C
C       Computes one fourth-order, five-point  Runge-Kutta             C
C       integration step for a second order differential               C
C       equation with first derivatives.  T is the independent         C
C       variable, X is the dependent variable, V is the                C
C       first derivative, and DT is the time step.  DERIVS is          C
C       the subroutine which calculates the second derivatives,        C
C       and the other variables are physical parameters needed         C
C       by DERIVS.  New  values are returned in T, X, and V.           C
C                                                                      C
C   PJP - revisions - 7/6/89                                           C
C                                                                      C
C       Added common blocks /ion/ and /rkz4/.
    C
C                                                                      C
C   PJP - revisions - 12/23/91                                         C
C                                                                      C
C       Removed CWY, PULSEX, and PULSEY from the common block          C
C       ION and from this subroutine.  Added SPNMAG to the ION         C
C       block.                                                         C
C                                                                      C
C   PJP - revisions - 1/16/92                                          C
C                                                                      C
C       Added AMFREQ, CWPHS2, CWPHS3, and CWPHS4 to the common         C
C       block ION.                                                     C
C       (Changed CWPHAS to CWPHS1.)
    C
C                                                                      C
C      PJP - revisions - 12/14/92                                      C
C                                                                      C
C              IRICE01 - IBM PC version.  Changed name of common block C
C              /rkz4/ to /rkzvar/.                                     C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC


      subroutine rkz4

      implicit none
      integer nvar,i
      parameter (nvar=6)
      double precision x(nvar),v(nvar),t,dt,k0(nvar),k1(nvar),
     1       k2(nvar),k3(nvar),k4(nvar),dv(nvar),xx(nvar),vv(nvar),
     2       tt,th4dx(nvar),th4dv(nvar),q,m,magmom,spin,v0,d,b0,b2,
     3       gradb,grade,cwfreq,cwphs1,gamma,spnmag,amfreq,cwphs2,
     4       cwphs3,cwphs4
     5       ,b1,b1freq,b1phas,b1angl
C
C     Common blocks (added on 7/6/89)
C
      common/ion/ q,m,magmom,spin,v0,d,b0,b2,gradb,grade,cwfreq,
     1       cwphs1,gamma,spnmag,amfreq,cwphs2,cwphs3,cwphs4
     5       ,b1,b1freq,b1phas,b1angl
      common/rkzvar/ t,x,v,dt

C
C     Calculate k0(i)
C
      tt = t
      do 10 i=1,nvar
             xx(i) = x(i)
             vv(i) = v(i)
 10   continue
```

```
      call derivs(tt,vv,dv,xx)
      do 15   i=1,nvar
              k0(i) = dt*dv(i)
 15   continue

C
C     Calculate k1(i)
C
      tt = t +  dt/2.
      do  20 i=1,nvar
              xx(i) = x(i) + dt*(4.*v(i) + k0(i))/8.
              vv(i) = v(i) + k0(i)/2.
 20   continue
      call derivs(tt,vv,dv,xx)
      do 25   i=1,nvar
              k1(i) = dt*dv(i)
 25   continue

C
C     Calculate  k2(i)
C
      do 30 i=1,nvar
              vv(i) = v(i) +  k1(i)/2.
 30   continue
      call derivs(tt,vv,dv,xx)
      do 35   i=1,nvar
              k2(i) = dt*dv(i)
 35   continue

C
C     Calculate  k3(i)
C
      tt= t + dt
      do 40 i=1,nvar
              xx(i) = x(i) + dt*(2.*v(i) + k2(i))/2.
              vv(i) = v(i) + k2(i)
 40   continue
      call derivs(tt,vv,dv,xx)
      do 45   i=1,nvar
              k3(i) = dt*dv(i)
 45   continue

C
C     If desired, calculate error terms
C
C     tt = t + dt*0.75
C     do 50 i=1,nvar
C             xx(i)=x(i)+dt*(48.*v(i)+7.*k0(i)+11.*k1(i))/64.
C             vv(i)=v(i)+(5.*k0(i)+7.*k1(i)+13.*k2(i)-k3(i))/32.
C50   continue
C     call derivs(tt,vv,dv,xx)
C     do 55   i=1,nvar
C             k4(i) = dt*dv(i)
C55   continue
C     do 57 i=1,nvar
C             th4dx(i)=dt*(2.*k0(i)-3.*(k1(i)+k2(i))+4.*k3(i))/9.
C             th4dv(i)=(2./3.)*(-1.*k0(i)+3.*(k1(i)+k2(i)+k3(i))
C    1              -8.*k4(i))
C57   continue

C
C     Calculate new T, X, V
C
      t = t + dt
      do 60 i=1,nvar
              x(i) = x(i) + dt*(v(i) + (k0(i)+k1(i)+k2(i))/6.)
              v(i) = v(i) + (k0(i) + 2*(k1(i)+k2(i)) + k3(i) )/6.
 60   continue


      return
      end



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                           C
C   DERIVS            Pedro J. Pizarro, 3/2/89                              C
C                                                                           C
C       Sets up the differential equations governing ion                    C
C       trajectories.  Edit this file to change electric                    C
C       and magnetic fields, as well as charge and ionic                    C
C       mass, in a given problem.                                           C
C                                                                           C
C   PJP - revisions - 7/6/89                                                C
C                                                                           C
C       Added common block /ion/.                                           C
C                                                                           C
C   PJP - revisions - 10/18/91                                              C
C                                                                           C
C       Programmed the space and spin equations for TSG.                    C
C       GRADB is now the TSG rf field gradient strength.                    C
C                                                                           C
C   PJP - revisions - 12/17/91                                              C
C                                                                           C
C       Changed DERIVS to calculate dM/dt from M cross B.                   C
C                                                                           C
C   PJP - revisions - 12/18/91                                              C
C                                                                           C
C       Store dM/dt in DERIVS as v(i),i=4,5,6.
    C
C                                                                           C
C   PJP - revisions - 12/22/91                                              C
C                                                                           C
C       Moved renormalization of dM/dt back to the main                     C
C       program, out of DERIVS.
    C
C                                                                           C
C   PJP - revisions - 12/23/91                                              C
C                                                                           C
C       Removed CWY, PULSEX, and PULSEY from the common block               C
C       ION and from this subroutine.  Added SPNMAG to the ION              C
```

```
C       block.                                                        C
C                                                                     C
C   PJP - revisions - 1/16/92                                         C
C                                                                     C
C       TRAJ2COIL.FOR - rf excitation is provided by two coils        C
C       oscillating at the rf frequency CWFREQ with phases            C
C       CWPHS1 and CWPHS3, and amplitude modulated at the             C
C       frequency AMFREQ with phases CWPHS2 and CWPHS4.               C
C       AMFREQ, CWPHS2, CWPHS3, and CWPHS4 have been added to         C
C       the common block ION.                                         C
C       (Changed CWPHAS to CWPHS1.)
   C
C       Now calculate spin dependent force by using the               C
C       divergence DBDX(3) of the magnetic field.                     C
C                                                                     C
C   PJP - revisions - 1/21/92                                         C
C                                                                     C
C       Corrected magnetic field.                                     C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine derivs(t,v,dvdt,x)

      implicit none
      integer n
      parameter (n=6)
      double precision t,q,m,spin,v(n),dvdt(n),x(n),e(n),b(n),
     1       v0,d,v0d,gradb,grade,magmom,bmag,b0,b2,rad2,spnfac,
     2       cwfreq,cwphs1,gamma,spnmag,amfreq,cwphs2,cwphs3,cwphs4
     5       ,b1,b1freq,b1phas,b1angl
      double precision dbdt(3),dmdt(3),dbdx(3)

C
C     Common block (added on 7/6/89)
C
      common/ion/ q,m,magmom,spin,v0,d,b0,b2,gradb,grade,cwfreq,
     1       cwphs1,gamma,spnmag,amfreq,cwphs2,cwphs3,cwphs4
     5       ,b1,b1freq,b1phas,b1angl

C
C     Calculate square of transverse radius, rad2 (in cm**2)
C
C     rad2 = x(1)*x(1) + x(2)*x(2)
C
C     Set up charge (multiple of e), ionic mass (in kg)
C            Note:   e = 1.6021892D-19 coulomb
C                    mass of electron = 9.109534D-31 kg
C                    mass of proton = 1.6726485D-27 kg
C                    spin = 1. for up, -1. for down (spin-1/2)
C                    "magmom" = magnetic moment
C                    electron magmom = 9.284832d-17 erg/tesla
C                    proton magmom = 1.4106171d-19 erg/tesla
C
C     Penning trap: v0 in volt, d in centimeter, b0 in tesla
C
      v0d = v0/(d*d)

C
C     Gradient fields scaled to reach a maximum percentage of the
C            static fields at distance d from the origin.
C            Units of "gradb" are tesla/centimeter.
C            Units of "grade" are volt/centimeter.
C
C     Set up fields: E (in volt/centimeter) and B (in tesla)
C            b(3) >> b(1),b(2)
C            b2 in tesla/centimeter**2
C
      b(1) = -gradb*x(1)*dcos(cwfreq*t+cwphs3)*dcos(amfreq*t+cwphs4)
     1      + b1*dcos(b1freq*t+b1phas)*dcos(b1angl)
      b(2) = gradb*x(2)*dcos(cwfreq*t+cwphs1)*dcos(amfreq*t+cwphs2)
     1      + b1*dcos(b1freq*t+b1phas)*dsin(b1angl)
      b(3) = b0 + gradb*x(3)*(
     1        dcos(cwfreq*t+cwphs3)*dcos(amfreq*t+cwphs4)
     2      - dcos(cwfreq*t+cwphs1)*dcos(amfreq*t+cwphs2) )
      bmag = dsqrt(b(1)**2 + b(2)**2 + b(3)**2)
      dbdt(1) = -gradb*v(1)*dcos(cwfreq*t+cwphs3)
     1                    *dcos(amfreq*t+cwphs4)
     2  + gradb*x(1)*(
     3      amfreq*dcos(cwfreq*t+cwphs3)*dsin(amfreq*t+cwphs4)
     4    + cwfreq*dsin(cwfreq*t+cwphs3)*dcos(amfreq*t+cwphs4) )
     5      - b1*b1freq*dsin(b1freq*t+b1phas)*dcos(b1angl)
      dbdt(2) = gradb*v(2)*dcos(cwfreq*t+cwphs1)
     1                    *dcos(amfreq*t+cwphs2)
     2  - gradb*x(2)*(
     3      amfreq*dcos(cwfreq*t+cwphs1)*dsin(amfreq*t+cwphs2)
     4    + cwfreq*dsin(cwfreq*t+cwphs1)*dcos(amfreq*t+cwphs2) )
     5      - b1*b1freq*dsin(b1freq*t+b1phas)*dsin(b1angl)
      dbdt(3) = gradb*v(3)*(
     1            dcos(cwfreq*t+cwphs3)*dcos(amfreq*t+cwphs4)
     2          - dcos(cwfreq*t+cwphs1)*dcos(amfreq*t+cwphs2) )
     3  + gradb*x(3)*(
     4    - amfreq*dcos(cwfreq*t+cwphs3)*dsin(amfreq*t+cwphs4)
     5    - cwfreq*dsin(cwfreq*t+cwphs3)*dcos(amfreq*t+cwphs4)
     6    + amfreq*dcos(cwfreq*t+cwphs1)*dsin(amfreq*t+cwphs2)
     7    + cwfreq*dsin(cwfreq*t+cwphs1)*dcos(amfreq*t+cwphs2) )
      dbdx(1) = -gradb*dcos(cwfreq*t+cwphs3)*dcos(amfreq*t+cwphs4)
      dbdx(2) =  gradb*dcos(cwfreq*t+cwphs1)*dcos(amfreq*t+cwphs2)
      dbdx(3) =  gradb*dcos(cwfreq*t+cwphs3)*dcos(amfreq*t+cwphs4)
     1        -gradb*dcos(cwfreq*t+cwphs1)*dcos(amfreq*t+cwphs2)


      e(1) = 0.5*v0d*x(1)
      e(2) = 0.5*v0d*x(2)
      e(3) = -1.0*v0d*x(3) - grade
C
C     Define spnfac
C
C     spnfac = (1.0d-3)*spin*magmom/(m*bmag)
C     spnfac = (1.0d-3)*gradb*dcos(cwfreq*t+cwphs1)/(2.d+0*m)
      spnfac = (1.0d-3)/(2.d+0*m)


C
C     Calculate dv/dt (in centimeter/sec^2) in each direction
```

```fortran
C
      dvdt(1) = (q/m)*(e(1)*1.0d4 + b(3)*v(2) - b(2)*v(3))
     1       + spnfac*x(4)*dbdx(1)
      dvdt(2) = (q/m)*(e(2)*1.0d4 + b(1)*v(3) - b(3)*v(1))
     1       + spnfac*x(5)*dbdx(2)
      dvdt(3) = (q/m)*(e(3)*1.0d4 + b(2)*v(1) - b(1)*v(2))
     1       + spnfac*x(6)*dbdx(3)
      dvdt(4) = gamma*( x(5)*dbdt(3) - x(6)*dbdt(2)
     1             + v(5)*b(3) - v(6)*b(2) )
      dvdt(5) = gamma*( x(4)*dbdt(3) - x(6)*dbdt(1)
     1             + v(4)*b(3) - v(6)*b(1) )*(-1.d+0)
      dvdt(6) = gamma*( x(4)*dbdt(2) - x(5)*dbdt(1)
     1             + v(4)*b(2) - v(5)*b(1) )

      return
      end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
C   QUANTS            Pedro J. Pizarro, 12/21/91                     C
C                                                                    C
C       Calculates cyclotron, magnetron, and axial quantum           C
C       numbers QUANTN, QUANTL, and QUANTK from positions and        C
C       velocities. QUANTN, QUANTK, and QUANTL are the only          C
C       variables altered by this subroutine.
   C
C                                                                    C
C      PJP - revisions - 12/14/92                                    C
C                                                                    C
C              IRICE01 - IBM PC version.  Changed name of common block C
C              /rkz4/ to /rkzvar/.                                   C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine quants

      implicit none
      integer nmax
      parameter (nmax=6)
      double precision x(nmax),v(nmax),m,wp,wm,wz,hbar,
     1       quantn,quantl,quantk,t,dt,qsign,pi
      common/rkzvar/ t,x,v,dt
      common/quantc/wp,wm,wz,hbar,quantn,quantl,quantk,qsign,m

      quantn=(m/(2.*hbar*(wp-wm)))*((v(1)-qsign*wm*x(2))**2 +
     1              (v(2)+qsign*wm*x(1))**2)*1.d-4
      quantl=(m/(2.*hbar*(wp-wm)))*((v(1)-qsign*wp*x(2))**2 +
     1              (v(2)+qsign*wp*x(1))**2)*1.d-4
      quantk=(m/(2.*hbar))*(wz*x(3)*x(3)+v(3)*v(3)/wz)*1.d-4

      return
      end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
C   PSHIFT            Pedro J. Pizarro, 12/23/91                     C
C                                                                    C
C       Calculates the instantaneous TSG rf phase, and outputs       C
C       a flag to the screen when the phase is shifted.              C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine pshift(cwphas,t)

      implicit none
      double precision cwphas,t,
     1            oldphs,cwphs0,phshft,dlshft,t1,phsfrq,pi
      common/rfphas/  oldphs,cwphs0,phshft,dlshft,t1,phsfrq,pi

      oldphs = cwphas
      cwphas = cwphs0 + phshft*dint( dlshft + (t-t1)*phsfrq )
      if(cwphas.ne.oldphs) write(6,*) 't = ',t,
     1       ' sec;  new phase = ',cwphas*180.d+0/pi,' deg'

      return
      end


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
C   RENORM            Pedro J. Pizarro, 12/23/91                     C
C                                                                    C
C       Renormalizes the spin vector and dM/dt.                      C
C                                                                    C
C   PJP - revisions - 1/16/92                                        C
C                                                                    C
C       TRAJ2COIL.FOR - rf excitation is provided by two coils       C
C       oscillating at the rf frequency CWFREQ with phases           C
C       CWPHS1 and CWPHS3, and amplitude modulated at the            C
C       frequency AMFREQ with phases CWPHS2 and CWPHS4.              C
C       AMFREQ, CWPHS2, CWPHS3, and CWPHS4 have been added to        C
C       the common block ION.                                        C
C       (Changed CWPHAS to CWPHS1.)
   C
C       Changed B(2) to B(3).                                        C
C                                                                    C
C   PJP - revisions - 1/21/92                                        C
C                                                                    C
C       Corrected magnetic field.                                    C
C                                                                    C
C      PJP - revisions - 12/14/92                                    C
C                                                                    C
C              IRICE01 - IBM PC version.  Changed name of common block C
C              /rkz4/ to /rkzvar/.                                   C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine renorm

      implicit none
      integer nmax
      parameter(nmax=6)
```

```
      double precision q,m,magmom,spin,v0,d,b0,b2,gradb,grade,
     1       cwfreq,cwphs1,gamma,spnmag,t,x(nmax),v(nmax),dt,
     2       b(3),spnorm,amfreq,cwphs2,cwphs3,cwphs4
     5       ,b1,b1freq,b1phas,b1angl
      common/ion/ q,m,magmom,spin,v0,d,b0,b2,gradb,grade,cwfreq,
     1       cwphs1,gamma,spnmag,amfreq,cwphs2,cwphs3,cwphs4
     5       ,b1,b1freq,b1phas,b1angl
      common/rkzvar/ t,x,v,dt

C
C     Renormalize spin vector M = (x(4),x(5),x(6))
C
      spnorm = dsqrt(x(4)**2 + x(5)**2 + x(6)**2)
      x(4) = spnmag*x(4)/spnorm
      x(5) = spnmag*x(5)/spnorm
      x(6) = spnmag*x(6)/spnorm
C
C     Renormalize dM/dt = (v(4),v(5),v(6))
C
      b(1) = -gradb*x(1)*dcos(cwfreq*t+cwphs3)*dcos(amfreq*t+cwphs4)
     1       + b1*dcos(b1freq*t+b1phas)*dcos(b1angl)
      b(2) = gradb*x(2)*dcos(cwfreq*t+cwphs1)*dcos(amfreq*t+cwphs2)
     1       + b1*dcos(b1freq*t+b1phas)*dsin(b1angl)
      b(3) = b0 + gradb*x(3)*(
     1         dcos(cwfreq*t+cwphs3)*dcos(amfreq*t+cwphs4)
     2       - dcos(cwfreq*t+cwphs1)*dcos(amfreq*t+cwphs2) )
      v(4) = gamma*(x(5)*b(3)-x(6)*b(2))
      v(5) = gamma*(x(4)*b(3)-x(6)*b(1))*(-1.d+0)
      v(6) = gamma*(x(4)*b(2)-x(5)*b(1))

      return
      end
```

Appendix B: Analytical Model Simulation Programs

The detailed study of ion motions and proposals to incorporate spin into the motional modes produced analytical approximations to many quantities of interest. In particular, closed form expressions were derived for the axial frequency in the presence of a magnetic bottle, the change in cyclotron radius due to SLIRICE, and other important quantities. These expressions can be assembled together into analytical model simulation programs that can simulate entire experimental sequences, from ion creation through detection of the proposed spin-dependent phenomena. Various versions of these programs were written to model several different ideas, including the experiments proposed here. The most recent of these simulations, AXIALIR8.FOR, is included in this appendix. It was used to model the SLIRICE experimental sequence proposed in Chapter 5.

The time sequence of the simulation is that proposed for the SLIRICE experiment. The mechanics of the simulation are explained in detail in the program comment lines. AXIALIR8 is written in FORTRAN and has been used with the Microsoft FORTRAN compiler for the DOS operating system on IBM PC compatible computers.

Also included in this appendix are all the supporting materials necessary to recreate the simulation figures presented at the end of Chapter 5. The parameter file Y7.PAR was copied to AXIAL.PAR before running AXIALIR8. The output file Y7.OUT includes program run information. After completion of the main program, the batch file Y7.BAT controlled runs of the program PLOTCUT3.FOR to isolate the simulation data range of interest. Final processing and graphics preparation were performed with Mathematica for DOS, using the input file Y7.SUM.

B.1 AXIALIR8.FOR

```
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c                                                                    c
c     AXIALTSG.FOR - Pedro J. Pizarro, 5/12/92                       c
c                                                                    c
c            Based on AXIALCV5_BRIEF.FOR.                            c
c            Creates an ensemble of ions with random positions and   c
c            Maxwell-Boltzmann velocities.  Ions are created in the  c
c            absence of a magnetic bottle by an electron beam along  c
c            the z-axis of radius RE, with axial potential           c
c            VINIT.  The potential is then changed in a time TV      c
c            to V0.  The cyclotron motion is excited via ICR, and    c
c            the ion cloud previously centered about the origin is   c
c            translated along the x-axis to DX with corresponding    c
c            change in the x velocity DVX; since the first           c
c            magnetic bottle period will dephase the cyclotron       c
c            coherence via frequency shifts, the ions are assigned   c
c            random cyclotron phases PHIP.  The magnetic bottle      c
c            is then turned on linearly over time TB2.  The axial    c
c            signal is recorded, and detection damping effects are   c
c            included.  The bottle is swept back to zero,            c
c            positive, and the axial amplitude and phase of each ion c
c            are adjusted accordingly.  A period of spin-dependent   c
c            cyclotron work via a transverse Stern-Gerlach effect    c
c            follows, and the new cyclotron radius is calculated.    c
c            The effects of an NMR period are simulated, and another c
c            TSG cyclotron radius change takes place.  The bottle is c
c            ramped back to B2.  The axial signal is recorded again. c
c            This entire sequence is repeated for varying NMR cw     c
c            irradiation times t1.  The period 1 signal is blanked   c
c            in the frequency domain by removing all points in the   c
c            FFT magnitude spectrum with an intensity less than the  c
c            sum of the mean plus THRESHold standard deviations.     c
c            For each t1, the period 1 signals are multiplied by the c
c            time-reversed period 2  signals.  The output data is    c
c            the average of the FFT magnitudes of the time-reversed  c
c            products.                                               c
c                                                                    c
c     Revisions - Pedro J. Pizarro, 5/13/92                          c
c                                                                    c
c            Added a switch to turn off noise: SNOISE is read from   c
c            the end of AXIAL.PAR.  If 0, it turns off noise; if     c
c            non-zero, it is set to 1.d+0, and noise is included.    c
c            Count number of ions which experience spin flips in     c
c            each t1 bin: the counter is FLIPS(IT1).                 c
c            Added filtering, turned on by the switch IFILTR, read   c
c            from the end of AXIAL.PAR.  If IFILTR=1, ions with      c
c            axial frequencies outside the detection bandwidth       c
c            (centered about the REFerence frequency) are not        c
c            included in the signal.  The number of ions included    c
c            in each t1 bin during the first detection period is     c
c            stored as FILTER(IT1).  The spectral half-width SW2 is  c
c            calculated for filtering.                               c
c            Note that the trapped ions are counted at the end of    c
c            SETUP.  The trapping flag can thus be reset by DETECT   c
c            to neglect ions outside the filter bandwidth.  In this  c
c            way, the spin flip count will include only ions within  c
c            the filter bandwidth.                                   c
c                                                                    c
c                                                                    c
c     Revisions - Pedro J. Pizarro, 5/15/92                          c
c                                                                    c
c            Corrected UMECH.                                        c
c                                                                    c
c     AXIALIR1 - Pedro J. Pizarro, 4/6/93                            c
c                                                                    c
c            DOS version.                                            c
c                                                                    c
c     AXIALIR4 - Pedro J. Pizarro, 4/20/93                           c
c                                                                    c
c            Added spin-locking field to IRICE.  See PJP notes of    c
c            12/29/92 for SLIRICE and 4/15/93 for spin flip          c
c            probabilities.  IONT1, FLIPS, and FILTER now have       c
c            dimension 1, not MAXT1.                                 c
c                                                                    c
c     AXIALIR5 - Pedro J. Pizarro, 5/10/93                           c
c                                                                    c
c            Resonance offset of each ion is now DWI(ION), randomly  c
c            set to be zero for half of the ions and DW for the      c
c            other half.  This is done in SETUP.                     c
c                                                                    c
c     AXIALIR7 - Pedro J. Pizarro, 5/12/93                           c
c                                                                    c
c            Based on AXIALIR7.  Corrected expressions for W1, the   c
c            spin-lock Rabi frequency.  See PJP notes of 5/12/93.    c
c                                                                    c
c     AXIALIR8 - Pedro J. Pizarro, 5/12/93                           c
c                                                                    c
c            Based on AXIALIR7.  Resonance offset of ions is now set c
c            to either DW or DW/2.  This is done in SETUP.           c
c                                                                    c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      program axialir8

      implicit none

      integer narrmx,maxion,maxt1
      parameter(narrmx=2048,maxion=1000,maxt1=40)

      real*8 q,m,u,b0,b2,v0,vinit,d,dx,hbar,z0,temp,wz,pi,pi2,
     1       wc,wp,wm,kb,p0,g,b1,w1,dw,dwi(maxion),mrres,
     2       umech(maxion),za(maxion),phiz(maxion),workr(2,narrmx),
```

```
     3       worki(2,narrmx),rmsv,kappa,gamma,
     4       vmbfac,vtable(narrmx),re,dts,dt1,sigres,ref,tb2,
     5       pp(maxion),pm(maxion),rdamp,tempr,tv,wzinit,rtrap,sw2,
     6       bbb,dvx,grad,tg,wzb2(maxion),phip(maxion),snoise,
     7       sigr(narrmx),sigi(narrmx),sigm(narrmx)
      real*4 rand
      integer i,j,npts,nions,ftrap(maxion),it1,idet,nvmb,nvtab,
     1       nt1,icount,idum,ishot,ioncnt,iont1,
     2       flips,filter,ifiltr
      integer*2 kran,ipad
      common/vars/q,m,u,b0,b2,v0,vinit,d,dx,hbar,z0,temp,wz,pi,pi2,
     1       wc,wp,wm,kb,p0,g,b1,w1,dw,dwi,mrres,umech,za,phiz,
     2       rmsv,kappa,gamma,vmbfac,vtable,re,dts,dt1,
     3       npts,nions,ftrap,it1,nt1,idet,nvmb,nvtab,sigres,ref,
     4       tb2,icount,kran,ipad,idum,ishot,ioncnt,workr,worki,
     5       pp,pm,rdamp,tempr,tv,wzinit,rtrap,sw2,
     6       bbb,dvx,grad,tg,wzb2,phip,flips,filter,ifiltr,snoise
      character*72 outfil
      integer nshots,ifor,iinv,non,noff
CIR3      real*8 workm(narrmx),thresh,dftmmx,sr(narrmx),si(narrmx)
      real*8 workm(narrmx),thresh,dftmmx
CIR3      real*4 bin(6,narrmx),ftm(2,narrmx)
      integer indx
      character*2 at1(maxt1)
      integer*2 ihr,imin,isec,i100th,iyr,imon,iday

      call gettim(ihr,imin,isec,i100th)
      call getdat(iyr,imon,iday)

      write(6,*) 'Beginning execution of program AXIALIR8...'

C
C     Constants:
C             hbar = 1.0545887d-34 J sec
C             q = 1.6021892d-19 coulomb
C             electron m = 9.109534d-31 kg
C             proton m = 1.6726485d-27 kg
C             electron u = 9.284832d-24 joule/tesla
C             proton u = 1.4106171d-26 joule/tesla
C             Boltzmann constant = kb = 1.38d-23 joule/molecule-K
C
C     pi = 3.14159 26535 89793 23846 26433 83279 50288 41971 69399 37511
C     pi2 = 2*pi
C
      hbar = 1.0545887d-34
      pi = 3.14159265358979323846264338327950d+0
      pi2 = 2.d+0*pi
      kb = 1.38d-23
      g = 2.d+0*1.001159657d+0
      kran = 1035
      call seed(kran)
      idum = -1001
      ifor = -1       !FFT direction (forward)
      iinv = 1        !FFT direction (inverse)
      non = 1         !B2SWEEP turn-on
      noff = -1       !B2SWEEP turn-off
C
C     Create t1 labels for file names
C
      open(unit=8,file='scratch.txt',status='new')
      do 1 i=1,maxt1
            write(8,2)i
 1    continue
 2    format(i2.2)
      close(unit=8)
      open(unit=8,file='scratch.txt',status='old')
      do 3 i=1,maxt1
            read(8,4) at1(i)
 3    continue
 4    format(a2)
      close(unit=8,status='delete')

C
C     Input:
C
      open(unit=8,file='axial.par',status='old')

      read(8,5) outfil
 5    format(a72)             !Output file name
      indx = index(outfil,' ')!INDX will be used to truncate OUTFIL
      if(indx.eq.0) indx=73   !    at beginning of blank characters
      indx = indx - 1         !    - now use OUTFIL(1:INDX)
      read(8,10) q            !Ion charge, in coulomb
      read(8,10) m            !Ion mass, in kg
      read(8,10) u            !Ion magnetic moment, in joule/tesla
      read(8,10) b0           !Magnetic field, in tesla
      read(8,10) b2           !Magnetic bottle, in tesla/(meter**2)
      b2 = dabs(b2)           !Positive B2
      read(8,10) vinit        !Initial trap potential, in volt
      read(8,10) v0           !Trap potential, in volt
      read(8,10) d            !Characteristic trap dimension, in cm
      d=d/100.d+0             !Convert d to meter
      read(8,10) z0           !Trap length, in cm
      z0=dabs(z0)/100.d+0     !Convert z0 to meter; positive
      rtrap = dsqrt(4.d+0*d*d - 2.d+0*z0*z0)  !Penning trap radius
      read(8,10) temp         !Ion temperature, in kelvin
 10   format(d30.15)
      read(8,10) tempr        !Detector temperature, in kelvin
      read(8,12) nions        !Number of ions generated
 12   format(i10)
      read(8,10) vmbfac       !Ions generated with v<=vmbfac*(mean v)
      read(8,12) nvmb         !Number of velocities in velocity table
      read(8,12) nvtab        !Number of entries in velocity table
      read(8,10) re           !Radius of ionizing electron beam, in um
      re = re*1.d-6           !Convert re to meter
      read(8,12) npts         !Number of time domain points for signal
      read(8,10) sigres       !Signal resolution (Hz)
      dts = 1.d+0/(sigres*dfloat(npts))       !Signal time step
      sw2 = 0.5d+0*sigres*dfloat(npts)*pi2    !Half-width in radians
      read(8,10) rdamp        !Signal detection resistance, in ohm
      read(8,10) ref          !Signal detection reference frequency (Hz)
```

```
      ref = pi2*ref             !Convert reference frequency to rad/sec
      read(8,12) nt1            !Number of points in t1 for MR
      read(8,10) mrres          !MR resolution (Hz)
      dt1 = 1.d+0/(mrres*dfloat(nt1)) !MR time step
      read(8,10) b1             !Spin lock field strength, in gauss
      b1 = b1*1.d-4             !Convert b1 to tesla
      read(8,10) dw             !MR resonance offset, in MHz
      dw = dw*1.d+6*pi2         !Convert dw to rad/sec
      read(8,10) tv             !Total time for Vinit to V0 sweep, in sec
      read(8,10) dx             !Change in x due to ICR excitation
      dx = dx/100.d+0           !dx, in m
      read(8,10) tb2            !Total time for B2 turn-on or turn-off, in sec
      read(8,10) grad           !TSG gradient strength, in T/m
      w1 = 0.5*g*u*b1/hbar      !Spin lock Rabi frequency, in rad/sec
      read(8,10) tg             !TSG excitation time, in sec
      read(8,12) nshots         !Number of shots for signal averaging
      read(8,10) thresh         !Number of standard deviations above mean
                                !FFT magnitude which are discarded by blanking
      read(8,10) snoise         !Noise switch
      if(snoise.ne.0.d+0) snoise=1.d+0        !SNOISE is 0. or 1.
      read(8,12) ifiltr         !Filtering switch

      close(unit=8)



C
C     Open output information file
C
      open(unit=10,file=outfil(1:indx)//'.out',
     1             status='new',form='formatted')

      kappa = 1.d+0             !Trap detection constant, dimensionless
      gamma = ((q*kappa/(2.d+0*z0))**2)*rdamp/m  !Signal damping, rad/sec
      rmsv = dsqrt(4.d+0*kb*tempr*rdamp*sigres*dfloat(npts))
                                          !Time-domain rms noise, V

      wzinit = dsqrt(q*vinit/(m*(d**2)))      !Initial wz, for Vinit
      wz = dsqrt(q*v0/(m*(d**2)))             !Final wz, for V0
      wc = q*b0/m
      wm = 0.5d+0*wc - 0.5d+0*dsqrt(wc*wc - 2.d+0*wz*wz)
      wp = wc-wm
      bbb = q*(v0-vinit)/(m*d*d*tv)   !Used for Vinit->V0 in SETUP
      dvx = wp*dx               !Change in vx due to ICR excitation

      write(6,*)'  **********  Program AXIALIR8  **********'
      write(10,*)'  **********  Program AXIALIR8  **********'
      write(6,*) 'charge q  = ',q,' coulomb'
      write(10,*) 'charge q  = ',q,' coulomb'
      write(6,*) 'mass m  = ',m,' kg'
      write(10,*) 'mass m  = ',m,' kg'
      write(6,*) 'spin magnetic moment u  = ',u,' joule/tesla'
      write(10,*) 'spin magnetic moment u  = ',u,' joule/tesla'
      write(6,*) 'b0 = ',b0,' tesla'
      write(10,*) 'b0 = ',b0,' tesla'
      write(6,*) 'b2 = ',b2,' gauss/square cm'
      write(10,*) 'b2 = ',b2,' gauss/square cm'
      write(6,*) 'initial trap potential = ',vinit,' volt'
      write(10,*) 'initial trap potential = ',vinit,' volt'
      write(6,*) 'v0 = ',v0,' volt'
      write(10,*) 'v0 = ',v0,' volt'
      write(6,*) 'trap characteristic dimension d  = ',d*100.,' cm'
      write(10,*) 'trap characteristic dimension d  = ',d*100.,' cm'
      write(6,*) 'trap length / 2 = z0 = ',z0*100.,' cm'
      write(10,*) 'trap length / 2 = z0 = ',z0*100.,' cm'
      write(6,*) 'trap radius = ',rtrap*100.,' cm'
      write(10,*) 'trap radius = ',rtrap*100.,' cm'
      write(6,*) 'axial frequency = ',dsqrt(q*v0/(m*(d**2)))/pi2,
     1       ' Hz'
      write(10,*) 'axial frequency = ',dsqrt(q*v0/(m*(d**2)))/pi2,
     1       ' Hz'
      write(6,*) 'ion temperature = ',temp,' K'
      write(10,*) 'ion temperature = ',temp,' K'
      write(6,*) 'detector temperature = ',tempr,' K'
      write(10,*) 'detector temperature = ',tempr,' K'
      write(6,*) 'number of ions generated = ',nions
      write(10,*) 'number of ions generated = ',nions
      write(6,*) 'ions generated with v <= mean v * ',vmbfac
      write(10,*) 'ions generated with v <= mean v * ',vmbfac
      write(6,*) 'target number of velocities in velocity table = ',
     1       nvmb
      write(10,*) 'target number of velocities in velocity table = ',
     1       nvmb
      write(6,*) 'number of entries in velocity table = ',nvtab
      write(10,*) 'number of entries in velocity table = ',nvtab
      write(6,*) 'radius of ionizing electron beam = ',re*1.d+6,' um'
      write(10,*) 'radius of ionizing electron beam = ',re*1.d+6,' um'
      write(6,*) 'number of time domain points for signal = ',npts
      write(10,*) 'number of time domain points for signal = ',npts
      write(6,*) 'signal resolution = ',sigres,' Hz'
      write(10,*) 'signal resolution = ',sigres,' Hz'
      write(6,*) 'signal time step = ',dts,' sec'
      write(10,*) 'signal time step = ',dts,' sec'
      write(6,*) 'signal detection resistance = ',rdamp,' ohm'
      write(10,*) 'signal detection resistance = ',rdamp,' ohm'
      write(6,*) 'signal detection damping = ',gamma/pi2,' Hz'
      write(10,*) 'signal detection damping = ',gamma/pi2,' Hz'
      write(6,*) 'signal reference frequency = ',ref/pi2,' Hz'
      write(10,*) 'signal reference frequency = ',ref/pi2,' Hz'
      write(6,*) 'signal bandwidth = ',sw2*2.d+0/pi2,' Hz'
      write(10,*) 'signal bandwidth = ',sw2*2.d+0/pi2,' Hz'
      write(6,*) 'rms noise voltage = ',rmsv,' V (total)'
      write(10,*) 'rms noise voltage = ',rmsv,' V (total)'
      write(6,*) 'rms noise voltage per FFT point = ',
     1       rmsv/dsqrt(dfloat(npts)),' V (total)'
      write(10,*) 'rms noise voltage per FFT point = ',
     1       rmsv/dsqrt(dfloat(npts)),' V (total)'
      write(6,*) 'number of points in t1 for MR = ',nt1
      write(10,*) 'number of points in t1 for MR = ',nt1
      write(6,*) 'MR resolution = ',mrres,' Hz'
      write(10,*) 'MR resolution = ',mrres,' Hz'
      write(6,*) 'MR time step = ',dt1,' sec'
      write(10,*) 'MR time step = ',dt1,' sec'
```

```
      write(6,*) 'Spin lock field strength = ',b1*1.d+4,' gauss'
      write(10,*) 'Spin lock field strength = ',b1*1.d+4,' gauss'
      write(6,*) 'MR resonance offset = (1/2,1)* ',dw*1.d-6/pi2,' MHz'
      write(10,*) 'MR resonance offset = (1/2,1)* ',dw*1.d-6/pi2,' MHz'
      write(6,*) 'total time for Vinit to V0 sweep = ',tv,' sec'
      write(10,*) 'total time for Vinit to V0 sweep = ',tv,' sec'
      write(6,*) 'voltage with axial amplitude z0/2 = ',
     1       0.5d+0*z0*kappa*(q*0.5d+0/z0)*wz*rdamp,' V (total)'
      write(10,*) 'voltage with axial amplitude z0/2 = ',
     1       0.5d+0*z0*kappa*(q*0.5d+0/z0)*wz*rdamp,' V (total)'
      write(6,*) 'change in x by ICR (ICR radius) = ',dx*100.,' cm'
      write(10,*) 'change in x by ICR (ICR radius) = ',dx*100.,' cm'
      write(6,*) 'total time for 0 to B2 sweep = ',tb2,' sec'
      write(10,*) 'total time for 0 to B2 sweep = ',tb2,' sec'
      write(6,*) 'TSG gradient strength = ',grad,' T/m'
      write(10,*) 'TSG gradient strength = ',grad,' T/m'
      write(6,*) 'TSG excitation time = ',tg,' sec'
      write(10,*) 'TSG excitation time = ',tg,' sec'
      write(6,*) 'Spin lock frequency = ',w1/pi2,' Hz'
      write(10,*) 'Spin lock frequency  = ',w1/pi2,' Hz'
      write(6,*) 'number of shots = ',nshots
      write(10,*) 'number of shots = ',nshots
      write(6,*) 'threshold (#standard deviations ',
     1       'above mean FFT magnitude '
      write(10,*) 'threshold (#standard deviations ',
     1       'above mean FFT magnitude '
      write(6,*) '           discarded by blanking) = ',thresh
      write(10,*) '           discarded by blanking) = ',thresh
      if(snoise.eq.0.d+0) then
            write(6,*)'snoise = ',snoise,'->no noise included'
            write(10,*)'snoise = ',snoise,'->no noise included'
      else
            write(6,*)'snoise = ',snoise,'->full noise included'
            write(10,*)'snoise = ',snoise,'->full noise included'
      endif
      if(ifiltr.eq.1) then
            write(6,*)'ifiltr = ',ifiltr,'->signal filter ON'
            write(10,*)'ifiltr = ',ifiltr,'->signal filter ON'
      else
            write(6,*)'ifiltr = ',ifiltr,'->signal filter OFF'
            write(10,*)'ifiltr = ',ifiltr,'->signal filter OFF'
      endif


C
C     Setup:  The flag ftrap(i) is 1 if ion i is
C                   still trapped or 0 if it is not.
C             MBDIST sets up the Maxwell-Boltzmann velocity grid.
C             SETUP assigns random x(i), y(i);
C                     Maxwell-Boltzmann vx(i), vy(i); if trapped,
C                     assigns random z(i); Maxwell-Boltzmann vz(i);
C                     if trapped, convert into za(i), phiz(i),
C                     and umech(i), which are returned to the
C                     main program.
C
      call mbdist
C
C     LOOP OVER T1 (INDEX IT1)
C
      do 130 it1=1,nt1
      write(6,*)'~***t1 bin # ',it1


C
C     Initialize the signal array.
C
            do 16 i=1,npts
                  sigm(i) = 0.d+0     !Final signal FT
CIR3                ftm(1,i) = 0.d+0    !Unblanked FT(S1)
CIR3                ftm(2,i) = 0.d+0    !Blanked FT(S1)
CIR3                bin(1,i) = 0.d+0    !Re(S1)
CIR3                bin(2,i) = 0.d+0    !Im(S1)
CIR3                bin(3,i) = 0.d+0    !Re(S2)
CIR3                bin(4,i) = 0.d+0    !Im(S2)
CIR3                bin(5,i) = 0.d+0    !Postblank Re(S1)
CIR3                bin(6,i) = 0.d+0    !Postblank Im(S1)
 16         continue
C
C     Initialize the ion counters
C
            iont1 = 0
            flips = 0
            filter = 0
C
C     LOOP OVER NSHOTS
C
      DO 100 ISHOT = 1,NSHOTS
      write(6,*)'~***signal shot # ',ishot
            do 20 i=1,nions
                  ftrap(i) = 1    !Initialize trapping flag
 20         continue

C
C     Initially, trapping voltage is VINIT.  This governs detrapping in
C     SETUP.  Trapping voltage is changed to V0 at the end of SETUP.
C
            call setup
            if(ioncnt.eq.0) goto 100        !Skip loop if no ions
            iont1 = iont1 + ioncnt
C
C     Turn on B2
C
            call b2sweep(non)
C
C     Detection period 1
C
C           write(6,*) '~detection period 1'
            idet = 1        !idet = 1 for period 1 signal
            call detect
C
C     Turn off B2
C
            call b2sweep(noff)
C
```

```
C     TSG - MR - TSG
C
              call tsg
C
C     Turn on B2
C
              call b2sweep(non)
C
C     Detection period 2
C
C             write(6,*) '~detection period 2'
              idet = 2        !idet = 2 for period 2 signal
              call detect
C
C     Multiply period 1 signal by the time-reversed period 2 signal.
C     Save all signals.
C
              do 40 i=1,npts
                     sigr(i) = workr(1,i)     !Save period 1 signal to be
                     sigi(i) = worki(1,i)     !blanked immediately
 40           continue
C
C     Transform, then blank out (PEAKM4) noise in the transform of period 1.
C     Note that is sufficient to blank out the noise from period 1 only.
C
CIR3              call fft_complex(sigr,sigi,npts,sr,si,ifor)
              call fft_complex(sigr,sigi,npts,sigr,sigi,ifor)
              dftmmx = 0.d+0
              do 50 i=1,npts
CIR3                     workm(i) = dsqrt(sr(i)**2+si(i)**2)      !Mag[FFT]
                     workm(i) = dsqrt(sigr(i)**2+sigi(i)**2)      !Mag[FFT]
CIR3                     ftm(1,i) = ftm(1,i) + sngl(workm(i))
                     dftmmx = dmax1(dftmmx,workm(i))
 50           continue
CIR3              call peakm5(narrmx,npts,workm,sr,si,thresh)
              call peakm5(narrmx,npts,workm,sigr,sigi,thresh)
CIR3              do 55 i=1,npts
CIR3                     ftm(2,i) = ftm(2,i) +
CIR3     1                       sngl(dsqrt(sr(i)**2+si(i)**2))
CIR3 55           continue
C
C     Inverse transform to get time domain signal for period 1
C
CIR3              call fft_complex(sr,si,npts,sr,si,iinv)
              call fft_complex(sigr,sigi,npts,sigr,sigi,iinv)
C
C     Calculate final time-reversed product signal, transform, and
C     add magnitude to averaging bin
C
              do 60 i=1,npts
                     j = npts - i + 1
CIR3                     sigr(i) = sr(i)*workr(2,j)
CIR3     1                             - si(i)*worki(2,j)
CIR3                     sigi(i) = si(i)*workr(2,j)
CIR3     1                             + sr(i)*worki(2,j)
                     workr(1,i) = sigr(i)
                     worki(1,i) = sigi(i)
                     sigr(i) = workr(1,i)*workr(2,j)
     1                             - worki(1,i)*worki(2,j)
                     sigi(i) = worki(1,i)*workr(2,j)
     1                             + workr(1,i)*worki(2,j)
 60           continue
              call fft_complex(sigr,sigi,npts,sigr,sigi,ifor)
              do 70 i=1,npts
                     sigm(i) = sigm(i) +
     1                      dsqrt(sigr(i)**2 + sigi(i)**2)
CIR3                  bin(1,i) = bin(1,i) + sngl(workr(1,i))
CIR3                  bin(2,i) = bin(2,i) + sngl(worki(1,i))
CIR3                  bin(3,i) = bin(3,i) + sngl(workr(2,i))
CIR3                  bin(4,i) = bin(4,i) + sngl(worki(2,i))
CIR3                  bin(5,i) = bin(5,i) + sngl(sr(i))
CIR3                  bin(6,i) = bin(6,i) + sngl(si(i))
 70           continue

 100  CONTINUE

      write(6,105) it1,iont1,filter,flips
      write(10,105) it1,iont1,filter,flips
 105  format(1x,'Ions in MR loop # ',i3,': ',i5,' trapped; ',i5,
     1       ' through filter; ',i5,' spin flips')

C
C     Save data
C
CIR3              open(unit=9,file=outfil(1:indx)//'m'//at1(it1)//'fm1.dat',
CIR3     1             status='new',form='formatted')
CIR3              do 111 i=1,npts
CIR3                     write(9,*)sngl(-0.5d+0*sigres*dfloat(npts) +
CIR3     1                      sigres*dfloat(i)),ftm(1,i)
CIR3 111          continue
CIR3              close(unit=9)
CIR3
CIR3              open(unit=9,file=outfil(1:indx)//'m'//at1(it1)//'fm2.dat',
CIR3     1             status='new',form='formatted')
CIR3             do 113 i=1,npts
CIR3                     write(9,*)sngl(-0.5d+0*sigres*dfloat(npts) +
CIR3     1                      sigres*dfloat(i)),ftm(2,i)
CIR3 113          continue
CIR3              close(unit=9)

              open(unit=9,file=outfil(1:indx)//'m'//at1(it1)//'fm.dat',
     1             status='new',form='formatted')
              do 115 i=1,npts
                     write(9,*)sngl(-0.5d+0*sigres*dfloat(npts) +
     1                      sigres*dfloat(i)),sngl(sigm(i))
 115          continue
              close(unit=9)

CIR3              open(unit=9,file=outfil(1:indx)//'m'//at1(it1)//'s1r.dat',
CIR3     1             status='new',form='formatted')
CIR3              do 120 i=1,npts
CIR3                     write(9,*) sngl(dts*dfloat(i-1+2-1)),
```

```
CIR3     1                      bin(1,i)
CIR3 120          continue
CIR3              close(unit=9)
CIR3
CIR3              open(unit=9,file=outfil(1:indx)//'m'//at1(it1)//'s1i.dat',
CIR3     1             status='new',form='formatted')
CIR3              do 121 i=1,npts
CIR3                     write(9,*) sngl(dts*dfloat(i-1+2-1)),
CIR3     1                      bin(2,i)
CIR3 121          continue
CIR3              close(unit=9)
CIR3
CIR3              open(unit=9,file=outfil(1:indx)//'m'//at1(it1)//'s2r.dat',
CIR3     1             status='new',form='formatted')
CIR3              do 122 i=1,npts
CIR3                     write(9,*) sngl(dts*dfloat(i-1+2-1)),
CIR3     1                      bin(3,i)
CIR3 122          continue
CIR3              close(unit=9)
CIR3
CIR3              open(unit=9,file=outfil(1:indx)//'m'//at1(it1)//'s2i.dat',
CIR3     1             status='new',form='formatted')
CIR3              do 123 i=1,npts
CIR3                     write(9,*) sngl(dts*dfloat(i-1+2-1)),
CIR3     1                      bin(4,i)
CIR3 123          continue
CIR3              close(unit=9)
CIR3
CIR3              open(unit=9,file=outfil(1:indx)//'m'//at1(it1)//'sbr.dat',
CIR3     1             status='new',form='formatted')
CIR3              do 124 i=1,npts
CIR3                     write(9,*) sngl(dts*dfloat(i-1+2-1)),
CIR3     1                      bin(5,i)
CIR3 124          continue
CIR3              close(unit=9)
CIR3
CIR3              open(unit=9,file=outfil(1:indx)//'m'//at1(it1)//'sbi.dat',
CIR3     1             status='new',form='formatted')
CIR3              do 125 i=1,npts
CIR3                     write(9,*) sngl(dts*dfloat(i-1+2-1)),
CIR3     1                      bin(6,i)
CIR3 125          continue
CIR3              close(unit=9)

 130  CONTINUE

      write(6,*)
      write(10,*)
      write(6,*)'--->>  Data saved in these formatted files:'
      write(10,*)'--->>  Data saved in these formatted files:'
CIR3      write(6,*) 'Re(S1) in ',outfil(1:indx)//'m**s1r.dat'
CIR3      write(10,*) 'Re(S1) in ',outfil(1:indx)//'m**s1r.dat'
CIR3      write(6,*) 'Im(S1) in ',outfil(1:indx)//'m**s1i.dat'
CIR3      write(10,*) 'Im(S1) in ',outfil(1:indx)//'m**s1i.dat'
CIR3      write(6,*) 'Mag[FFT(S1)] in ',outfil(1:indx)//'m**fm1.dat'
CIR3      write(10,*) 'Mag[FFT(S1)] in ',outfil(1:indx)//'m**fm1.dat'
CIR3      write(6,*) 'Blanked Mag[FFT(S1)] in ',
CIR3     1       outfil(1:indx)//'m**fm2.dat'
CIR3      write(10,*) 'Blanked Mag[FFT(S1)] in ',
CIR3     1       outfil(1:indx)//'m**fm2.dat'
CIR3      write(6,*) 'Re(Inv FFT) of blanked FFT(S1) in ',
CIR3     1       outfil(1:indx)//'m**sbr.dat'
CIR3      write(10,*) 'Re(Inv FFT) of blanked FFT(S1) in ',
CIR3     1       outfil(1:indx)//'m**sbr.dat'
CIR3      write(6,*) 'Im(Inv FFT) of blanked FFT(S1) in ',
CIR3     1       outfil(1:indx)//'m**sbi.dat'
CIR3      write(10,*) 'Im(Inv FFT) of blanked FFT(S1) in ',
CIR3     1       outfil(1:indx)//'m**sbi.dat'
CIR3      write(6,*) 'Re(S2) in ',outfil(1:indx)//'m**s2r.dat'
CIR3      write(10,*) 'Re(S2) in ',outfil(1:indx)//'m**s2r.dat'
CIR3      write(6,*) 'Im(S2) in ',outfil(1:indx)//'m**s2i.dat'
CIR3      write(10,*) 'Im(S2) in ',outfil(1:indx)//'m**s2i.dat'
      write(6,*) 'Mag[FFT(blanked time-reversed product)] in ',
     1       outfil(1:indx)//'m**fm.dat'
      write(10,*) 'Mag[FFT(blanked time-reversed product)] in ',
     1       outfil(1:indx)//'m**fm.dat'

      write(6,*)
      write(10,*)
      write(6,200) imon,iday,iyr,ihr,imin,isec,i100th
      write(10,200) imon,iday,iyr,ihr,imin,isec,i100th
      call getdat(iyr,imon,iday)
      call gettim(ihr,imin,isec,i100th)
      write(6,202) imon,iday,iyr,ihr,imin,isec,i100th
      write(10,202) imon,iday,iyr,ihr,imin,isec,i100th
 200  format(1x,'Program started at: ',i2,'-',i2.2,'-',i4,
     1    1x,i2,':',i2.2,':',i2.2,'.',i2.2)
 202  format(1x,'Program ended at:   ',i2,'-',i2.2,'-',i4,
     1    1x,i2,':',i2.2,':',i2.2,'.',i2.2)

      close(unit=10)

      stop
      end


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     MBDIST - Pedro J. Pizarro, 1/8/91                                C
C                                                                      C
C             Creates a table of NVMB velocities from zero to VMBFAC   C
C             times the mean velocity with NVTAB entries.  The number  C
C             of times a given velocity is written to the table is     C
C             set to follow Maxwell-Boltzmann statistics.              C
C                                                                      C
C     Revisions - Pedro J. Pizarro, 5/13/92                            C
C                                                                      C
C             Altered common blocks and added use of WZB2(I) to make   C
C             the subroutine compatible with AXIALTSG.FOR.             C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

```
      subroutine mbdist

      implicit none

      integer narrmx,maxion,maxt1
      parameter(narrmx=2048,maxion=1000,maxt1=40)

      real*8 q,m,u,b0,b2,v0,vinit,d,dx,hbar,z0,temp,wz,pi,pi2,
     1       wc,wp,wm,kb,p0,g,b1,w1,dw,dwi(maxion),mrres,
     2       umech(maxion),za(maxion),phiz(maxion),workr(2,narrmx),
     3       worki(2,narrmx),rmsv,kappa,gamma,
     4       vmbfac,vtable(narrmx),re,dts,dt1,sigres,ref,tb2,
     5       pp(maxion),pm(maxion),rdamp,tempr,tv,wzinit,rtrap,sw2,
     6       bbb,dvx,grad,tg,wzb2(maxion),phip(maxion),snoise,
     7       sigr(narrmx),sigi(narrmx),sigm(narrmx)
      real*4 rand
      integer i,j,npts,nions,ftrap(maxion),it1,idet,nvmb,nvtab,
     1       nt1,icount,idum,ishot,ioncnt,iont1,
     2       flips,filter,ifiltr
      integer*2 kran,ipad
      common/vars/q,m,u,b0,b2,v0,vinit,d,dx,hbar,z0,temp,wz,pi,pi2,
     1       wc,wp,wm,kb,p0,g,b1,w1,dw,dwi,mrres,umech,za,phiz,
     2       rmsv,kappa,gamma,vmbfac,vtable,re,dts,dt1,
     3       npts,nions,ftrap,it1,nt1,idet,nvmb,nvtab,sigres,ref,
     4       tb2,icount,kran,ipad,idum,ishot,ioncnt,workr,worki,
     5       pp,pm,rdamp,tempr,tv,wzinit,rtrap,sw2,
     6       bbb,dvx,grad,tg,wzb2,phip,flips,filter,ifiltr,snoise

      real*8 vmean,dv,vv,f
      integer n

      icount = 0
      vmean = dsqrt(2.d+0*kb*temp/m)
      dv = vmbfac*vmean/(dfloat(nvmb))
      write(6,*) 'Mean ion velocity =        ',vmean,' m/sec'
      write(10,*) 'Mean ion velocity =        ',vmean,' m/sec'
      write(6,*) 'Velocity table increment = ',dv,' m/sec'
      write(10,*) 'Velocity table increment = ',dv,' m/sec'

      do 20 i=1,nvmb
            vv = dv*float(i)
            f = dsqrt(m/(pi2*kb*temp))*
     1            dexp(-0.5d+0*m*vv*vv/(kb*temp))
            n = nint(2.d+0*dfloat(nvtab)*f*vmbfac*vmean
     1            /dfloat(nvmb))
            do 10 j=icount+1,icount+n
                  vtable(j) = vv
10          continue
            icount = icount + n
20    continue

      write(6,*)'Number of entries in velocity table = ',icount
      write(10,*)'Number of entries in velocity table = ',icount


      return
      end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SETUP - Pedro J. Pizarro, 5/13/92                                C
C                                                                      C
C             Sets up initial radial (transverse) conditions for       C
C             ions.  Checks ion trapping.                              C
C             Sets up initial axial conditions for ions.  Checks ion   C
C             trapping.                                                C
C             Calculates change in phase due to Vinit -> V0; see       C
C             notes of 1/31/91 for details and definitions of BBB,     C
C             WZ0, and WZI.                                            C
C             Excites the cyclotron motion via ICR.  Checks ion        C
C             trapping.                                                C
C                                                                      C
C     AXIALIR5 - Pedro J. Pizarro, 5/10/93                             C
C                                                                      C
C             Resonance offset of each ion is now DWI(ION), randomly   C
C             set to be zero for half of the ions and DW for the       C
C             other half.  This is done in SETUP.                      C
C                                                                      C
C     AXIALIR8 - Pedro J. Pizarro, 5/12/93                             C
C                                                                      C
C             Based on AXIALIR7.  Resonance offset of ions is now set  C
C             to either DW or DW/2.  This is done in SETUP.            C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine setup

      implicit none

      integer narrmx,maxion,maxt1
      parameter(narrmx=2048,maxion=1000,maxt1=40)

      real*8 q,m,u,b0,b2,v0,vinit,d,dx,hbar,z0,temp,wz,pi,pi2,
     1       wc,wp,wm,kb,p0,g,b1,w1,dw,dwi(maxion),mrres,
     2       umech(maxion),za(maxion),phiz(maxion),workr(2,narrmx),
     3       worki(2,narrmx),rmsv,kappa,gamma,
     4       vmbfac,vtable(narrmx),re,dts,dt1,sigres,ref,tb2,
     5       pp(maxion),pm(maxion),rdamp,tempr,tv,wzinit,rtrap,sw2,
     6       bbb,dvx,grad,tg,wzb2(maxion),phip(maxion),snoise,
     7       sigr(narrmx),sigi(narrmx),sigm(narrmx)
      real*4 rand
      integer i,j,npts,nions,ftrap(maxion),it1,idet,nvmb,nvtab,
     1       nt1,icount,idum,ishot,ioncnt,iont1,
     2       flips,filter,ifiltr
      integer*2 kran,ipad
      common/vars/q,m,u,b0,b2,v0,vinit,d,dx,hbar,z0,temp,wz,pi,pi2,
     1       wc,wp,wm,kb,p0,g,b1,w1,dw,dwi,mrres,umech,za,phiz,
     2       rmsv,kappa,gamma,vmbfac,vtable,re,dts,dt1,
     3       npts,nions,ftrap,it1,nt1,idet,nvmb,nvtab,sigres,ref,
     4       tb2,icount,kran,ipad,idum,ishot,ioncnt,workr,worki,
     5       pp,pm,rdamp,tempr,tv,wzinit,rtrap,sw2,
     6       bbb,dvx,grad,tg,wzb2,phip,flips,filter,ifiltr,snoise
```

```
      real*8 x,y,vx,vy,rp,rm,z,vz,zamin,zamax

      ioncnt = 0      !Initialize trapped ion counter
      zamin = z0      !Initialize minimum amplitude
      zamax = 0.d+0   !Initialize maximum amplitude

C
C     Loop over all ions
C
      do 100 i=1,nions
C     write(6,*)'~SETUP: ion # ',i
C
C     Set up transverse coordinates
C
            call random(rand)
            x = 2.d+0*re*(dfloat(rand)-0.5d+0)
            call random(rand)
            j = nint(dfloat(icount-1)*dfloat(rand))+1
            vx = vtable(j)
            call random(rand)
            j = nint(dfloat(rand))
            vx = vx*(-1**j)
            call random(rand)
            y = 2.d+0*re*(dfloat(rand)-0.5d+0)
            call random(rand)
            j = nint(dfloat(icount-1)*dfloat(rand))+1
            vy = vtable(j)
            call random(rand)
            j = nint(dfloat(rand))
            vy = vy*(-1**j)
C
C     Convert to cyclotron and magnetron radii
C
            rp = dsqrt( (vx*vx + vy*vy + wm*wm*(x*x+y*y)
     1             + 2.d+0*wm*(x*vy-y*vx))/((wp-wm)**2) )
            rm = dsqrt( (vx*vx + vy*vy + wp*wp*(x*x+y*y)
     1             + 2.d+0*wp*(x*vy-y*vx))/((wp-wm)**2) )

            if(rp+rm .ge. rtrap) then
                  ftrap(i) = 0    !Flag goes to zero if detrapped
                  goto 100
            endif
C           write(6,*) '~ x, y=',x,y
C           write(6,*) '~vx,vy=',vx,vy
C           write(6,*) '~rp,rm=',rp,rm
C
C     Set up axial coordinates, with axial potential Vinit
C
            call random(rand)
            z = 2.d+0*z0*(dfloat(rand)-0.5d+0)
            call random(rand)
            j = nint(dfloat(icount-1)*dfloat(rand))+1
            vz = vtable(j)
            call random(rand)
            j = nint(dfloat(rand))
            vz = vz*(-1**j)
            za(i) = dsqrt(z*z + vz*vz/(wzinit*wzinit)) !Axial amplitude
            phiz(i) = atan(wzinit*z/vz)     !Axial phase if z is sine
            if(za(i) .ge. z0) then
                  ftrap(i) = 0              !Flag = 0 if detrapped
                  goto 100                  !Neglect detrapped ions
            endif
C           write(6,*) '~z,vz=',z,vz
C           write(6,*) '~za,phiz=',za(i),phiz(i)
C
C     Change trapping voltage from VINIT to V0:
C     use conservation of action to get new axial amplitude; calculate
C     new phase
C
            za(i) = za(i)*dsqrt(wzinit/wz)  !New amplitude
            if(za(i) .ge. z0) then
                  ftrap(i) = 0              !Flag = 0 if detrapped
                  goto 100                  !Neglect detrapped ions
            endif

            phiz(i) = phiz(i) +             !New phase
     1            (2.d+0/(3.d+0*bbb))*(wz**3 - wzinit**3)
C           write(6,*) '~v0: za,phiz=',za(i),phiz(i)
C
C     Excite cyclotron motion via ICR
C
            x = x + dx
            vx = vx + dvx
C
C     Convert to cyclotron and magnetron radii
C
            rp = dsqrt( (vx*vx + vy*vy + wm*wm*(x*x+y*y)
     1             + 2.d+0*wm*(x*vy-y*vx))/((wp-wm)**2) )
            rm = dsqrt( (vx*vx + vy*vy + wp*wp*(x*x+y*y)
     1             + 2.d+0*wp*(x*vy-y*vx))/((wp-wm)**2) )
            umech(i) = 0.5d+0*q*(wp*rp*rp+wm*rm*rm)       !mechanical moment

            pp(i) = rp
            pm(i) = rm

            if(rp+rm .ge. rtrap) then
                  ftrap(i) = 0    !Flag goes to zero if detrapped
                  goto 100
            endif
C
C     Set up random cyclotron phases
C
            call random(rand)
            phip(i) = pi2*dble(dfloat(rand))
C           write(6,*) '~ICR: rp,rm,phip=',pp(i),pm(i),phip(i)
C
C     Set up resonance offset
C
            call random(rand)
            dwi(i) = dw*0.5d+0*(1.d+0+dfloat(nint(dfloat(rand))))
                                                        !dw*(0.5 or 1)
```

```
C
C     Count number of trapped ions
C
              ioncnt = ioncnt + 1

              zamin = dmin1(zamin,za(i))       !Minimum amplitude
              zamax = dmax1(zamax,za(i))       !Maximum amplitude

 100  continue

C     if(ioncnt.eq.0) goto 200        !Don't write amplitudes if no ions

C     write(6,*) zamin*1.d+2,' <= axial amplitudes <= ',zamax*1.d+2,
C    1       ' cm'
C     write(6,*) zamin*kappa*q*wz*rdamp/(dsqrt(8.d+0)*z0),
C    1       ' <= rms axial voltage <= ',
C    2       zamax*kappa*q*wz*rdamp/(dsqrt(8.d+0)*z0),' V'

 200  return
      end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     DETECT - Pedro J. Pizarro, 1/9/91                                C
C                                                                      C
C              Calculates ion signal due to axial detection.  Adjusts  C
C              final axial amplitudes to reflect damping.              C
C                                                                      C
C     Revisions - Pedro J. Pizarro, 1/17/91                            C
C                                                                      C
C              No longer uses sig(,,).  Now uses work(,).              C
C                                                                      C
C     Revisions - Pedro J. Pizarro, 1/29/91                            C
C                                                                      C
C              Since now calculate a complex signal, replaced WORK(,)  C
C              with WORKI(,) and WORKI(,), the real and imaginary      C
C              parts of the signal, respectively.  WORKI lags 90       C
C              degrees behind WORKR, since WORKR is a sine and WORKI   C
C              is a negative cosine.                                   C
C                                                                      C
C     Revisions - Pedro J. Pizarro, 1/31/91                            C
C                                                                      C
C              Now use WZ2 (not WZ2-REF) to scale the signal           C
C              intensity.  Use voltage detection.                      C
C                                                                      C
C     Revisions - Pedro J. Pizarro, 5/13/92                            C
C                                                                      C
C              Altered common blocks and added use of WZB2(I) to make  C
C              the subroutine compatible with AXIALTSG.FOR.            C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine detect

      implicit none

      integer narrmx,maxion,maxt1
      parameter(narrmx=2048,maxion=1000,maxt1=40)

      real*8 q,m,u,b0,b2,v0,vinit,d,dx,hbar,z0,temp,wz,pi,pi2,
     1       wc,wp,wm,kb,p0,g,b1,w1,dw,dwi(maxion),mrres,
     2       umech(maxion),za(maxion),phiz(maxion),workr(2,narrmx),
     3       worki(2,narrmx),rmsv,kappa,gamma,
     4       vmbfac,vtable(narrmx),re,dts,dt1,sigres,ref,tb2,
     5       pp(maxion),pm(maxion),rdamp,tempr,tv,wzinit,rtrap,sw2,
     6       bbb,dvx,grad,tg,wzb2(maxion),phip(maxion),snoise,
     7       sigr(narrmx),sigi(narrmx),sigm(narrmx)
      real*4 rand
      integer i,j,npts,nions,ftrap(maxion),it1,idet,nvmb,nvtab,
     1       nt1,icount,idum,ishot,ioncnt,iont1,
     2       flips,filter,ifiltr
      integer*2 kran,ipad
      common/vars/q,m,u,b0,b2,v0,vinit,d,dx,hbar,z0,temp,wz,pi,pi2,
     1       wc,wp,wm,kb,p0,g,b1,w1,dw,dwi,mrres,umech,za,phiz,
     2       rmsv,kappa,gamma,vmbfac,vtable,re,dts,dt1,
     3       npts,nions,ftrap,it1,nt1,idet,nvmb,nvtab,sigres,ref,
     4       tb2,icount,kran,ipad,idum,ishot,ioncnt,workr,worki,
     5       pp,pm,rdamp,tempr,tv,wzinit,rtrap,sw2,
     6       bbb,dvx,grad,tg,wzb2,phip,flips,filter,ifiltr,snoise

      real*8 t,wz2,wzmin,wzmax,root2
      real*4 gasdev
      external gasdev

      root2 = dsqrt(2.d+0)
      wzmax = -1.d+0
      wzmin = 1.7d+38


C
C     Initialize the signal array.
C
      do 16 i=1,npts
             workr(idet,i) = 0.d+0
             worki(idet,i) = 0.d+0
 16   continue

      do 100 i=1,nions
             if(ftrap(i).eq.0) goto 100      !Neglect detrapped ions
C
C     Calculate signal.  Note that ref is subtracted from the axial
C     frequency.  This is equivalent to analog mixing with the
C     reference frequency followed by low-pass filtering (single
C     sideband mixing).  Note that the process reduces the signal
C     amplitude by a factor of the square root of two; a further factor of
C     the square root of two comes from the fact that we have split the
C     signal into two quadrature components, so the total factor is two.
C     The axial motion is assumed to behave as a sine, so the velocity
C     is a cosine, and the signal voltage is a sine.
C     AXIALCV2_NONOISE: FIRST IDET=1 DATA POINT IS ACQUIRED AT TIME
C     T=DTS, SO THAT LAST IDET=1 POINT COINCIDES WITH FIRST IDET=2 POINT
C     IN THE LIMIT TB2=0.
```

```
C
            wzmin = dmin1(wzmin,wzb2(i))
            wzmax = dmax1(wzmax,wzb2(i))
            wz2 = wzb2(i) - ref             !Single sideband mixing
C           write(6,*) '~ion # ',i,' (wz-ref)/2pi=',wz2/pi2
C
C     Filter signal
C
            if(ifiltr.eq.1) then
                    if(dabs(wz2).ge.sw2) then
                            ftrap(i) = 0
                            goto 100
                    endif
            endif
            if(idet.eq.1) filter = filter+1  !Filter counter

            do 50 j=1,npts
                    t = dts*dfloat(j-1+2-IDET)
                    workr(idet,j) = workr(idet,j) +
     1                      kappa*(q*0.5d+0/z0)*(wz2+ref)    !Factors
     2                      *(za(i)/2.d+0)*dexp(-1.d+0*gamma*t)      !Amp.
     3                      *dsin(wz2*t+phiz(i))                    !Freq.
                    worki(idet,j) = worki(idet,j) - !Quadrature signal
     1                      kappa*(q*0.5d+0/z0)*(wz2+ref)    !Factors
     2                      *(za(i)/2.d+0)*dexp(-1.d+0*gamma*t)      !Amp.
     3                      *dcos(wz2*t+phiz(i))                    !Freq.
 50         continue
C
C     Record final axial amplitude.  Note that, for a detection
C     period of length npts*dts, the first data point is acquired at
C     time t=0, the last point at t=(npts-1)*dts, and the final
C     axial amplitude is set at t=npts*dt.
C     AXIALCV2_NONOISE: FIRST IDET=1 DATA POINT IS ACQUIRED AT TIME
C     T=DTS, SO THAT LAST IDET=1 POINT COINCIDES WITH FIRST IDET=2 POINT
C     IN THE LIMIT TB2=0.
C
            t = dts*dfloat(npts)
            za(i) = za(i)*dexp(-1.d+0*gamma*t)
 100  continue

C
C     Final signal voltage is current*rdamp with noise voltage added.
C     Note that we add half the noise current to each channel.
C     The noise is Gaussian, from the function GASDEV, and with
C     rms value RMSV.
C
      do 150 j = 1,npts
            workr(idet,j) = workr(idet,j)*rdamp
     1              + snoise*rmsv*dble(gasdev(idum))/2.d+0
            worki(idet,j) = worki(idet,j)*rdamp
     1              + snoise*rmsv*dble(gasdev(idum))/2.d+0
 150  continue
C
C     Write axial frequency range to screen.
C
C     write(6,200) idet,wzmin/pi2,wzmax/pi2,(wzmax-wzmin)/pi2
 200  format(1x,'Period ',i1,': ',f16.5,' <=vz<=',
     1       f16.5,'; range = ',f10.5,' Hz')

      return
      end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                           C
C     B2SWEEP - Pedro J. Pizarro, 5/13/92                                   C
C                                                                           C
C             The switch NONOFF is 1 for turn-on and -1 for turn-off.       C
C             Randomly assigns SPIN to be 1 or -1 for B2 turn-on.           C
C             This simulates the projection of spin onto eigenvalues        C
C             quantized along the z-axis when the axial frequency is        C
C             measured in the presence of B2.                               C
C             The axial frequency in the presence of B2, WZB2(J), is        C
C             calculated for B2 turn-on, and is made                        C
C             available through the common block VARS for use by the        C
C             subroutine DETECT.                                            C
C             Calculates the new axial amplitude and phase after B2         C
C             is turned on or off adiabatically and linearly in time.       C
C             See 5/12/92 notes for closed-form integration used.           C
C             Since the experimental sequence of AXIALTSG starts off        C
C             with B2=0 when the ions are created and only later goes       C
C             to B2>0, the B2 turn on leads to an increase in the           C
C             axial frequency with a corresponding decrease in the          C
C             axial amplitude, and there is thus no need to check for       C
C             ion detrapping in this subroutine.                            C
C                                                                           C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine b2sweep(nonoff)

      implicit none

      integer narrmx,maxion,maxt1,nmax,nonoff
      parameter(narrmx=2048,maxion=1000,maxt1=40)

      real*8 q,m,u,b0,b2,v0,vinit,d,dx,hbar,z0,temp,wz,pi,pi2,
     1       wc,wp,wm,kb,p0,g,b1,w1,dw,dwi(maxion),mrres,
     2       umech(maxion),za(maxion),phiz(maxion),workr(2,narrmx),
     3       worki(2,narrmx),rmsv,kappa,gamma,
     4       vmbfac,vtable(narrmx),re,dts,dt1,sigres,ref,tb2,
     5       pp(maxion),pm(maxion),rdamp,tempr,tv,wzinit,rtrap,sw2,
     6       bbb,dvx,grad,tg,wzb2(maxion),phip(maxion),snoise,
     7       sigr(narrmx),sigi(narrmx),sigm(narrmx)
      real*4 rand
      integer i,j,npts,nions,ftrap(maxion),it1,idet,nvmb,nvtab,
     1       nt1,icount,idum,ishot,ioncnt,iont1,
     2       flips,filter,ifiltr
      integer*2 kran,ipad
      common/vars/q,m,u,b0,b2,v0,vinit,d,dx,hbar,z0,temp,wz,pi,pi2,
     1       wc,wp,wm,kb,p0,g,b1,w1,dw,dwi,mrres,umech,za,phiz,
     2       rmsv,kappa,gamma,vmbfac,vtable,re,dts,dt1,
     3       npts,nions,ftrap,it1,nt1,idet,nvmb,nvtab,sigres,ref,
     4       tb2,icount,kran,ipad,idum,ishot,ioncnt,workr,worki,
```

```
     5       pp,pm,rdamp,tempr,tv,wzinit,rtrap,sw2,
     6       bbb,dvx,grad,tg,wzb2,phip,flips,filter,ifiltr,snoise

      real*8 phase,t,aa,bb,spin
C
C     Integrate over linear bottle sweep.  See PJP notes from 1/18/91,
C     1/19/91.
C
      IF(NONOFF.EQ.1) GOTO 10
      IF(NONOFF.EQ.-1) GOTO 200
      write(6,*)'ERROR IN FLAG FOR SUBROUTINE B2SWEEP!!!  ',
     1       'NEED NONOFF = 1 or -1.'
      write(10,*)'ERROR IN FLAG FOR SUBROUTINE B2SWEEP!!!  ',
     1       'NEED NONOFF = 1 or -1.'
      GOTO 400

 10   do 100 j=1,nions
             if(ftrap(j).eq.0) goto 100      !Neglect detrapped ions
C
C     Set up spin
C
             call random(rand)
             spin = dfloat(nint(
     1              2.d+0*(dfloat(nint(dfloat(rand)))-0.5d+0))) !-1 or 1
C
C     New phase
C
             aa = wz*wz
             bb = (b2/(m*tb2))*(2.d+0*umech(j)-spin*2.d+0*u)
             phase = (2.d+0/(3.d+0*bb))*((aa+bb*tb2)**(1.5d+0)
     1              - aa**1.5d+0 )  !0 -> +B2

             phiz(j) = phiz(j) + phase              !New phase

             wzb2(j) = dsqrt(wz*wz + 2.d+0*(umech(j)*b2/m)
     1              - spin*(2.d+0*u*b2/m))       !+B2 frequency
C
C     Use conservation of action to get new axial amplitude
C
             za(j) = za(j)*dsqrt(wz/wzb2(j)) !New amplitude

 100  continue

      goto 400

 200  do 300 j=1,nions
             if(ftrap(j).eq.0) goto 300      !Neglect detrapped ions

             aa = wz*wz +
     1              (b2/m)*(2.d+0*umech(j)-spin*2.d+0*u)
             bb = (-1.d+0*b2/(m*tb2))*
     1              (2.d+0*umech(j)-spin*2.d+0*u)
             phase = (2.d+0/(3.d+0*bb))*((aa+bb*tb2)**(1.5d+0)
     1              - aa**1.5d+0 )  !+B2 -> 0

             phiz(j) = phiz(j) + phase              !New phase
C
C     Use conservation of action to get new axial amplitude
C
             za(j) = za(j)*dsqrt(wzb2(j)/wz) !New amplitude

 300  continue

 400  return
      end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     TSG - Pedro J. Pizarro, 5/13/92                                  C
C                                                                      C
C             Checks for ion trapping.  Reassigns SPIN(I) randomly to  C
C             1 or -1 to simulate the projection of spins previously   C
C             quantized along the z-axis (when B2 was on and the       C
C             axial frequency was detected) onto the transverse        C
C             eigenstates of the 2-coil TSG field.  Calculates the     C
C             change in cyclotron radius due to a TSG period with      C
C             gradient field strength GRAD over time TG exactly on     C
C             resonance; see 5/12/92 notes for equation.  Flips the    C
C             spin with probability given by the Rabi equation.        C
C             Calculates the change in cyclotron radius due to a       C
C             second TSG period.  Recalculates the ion's mechanical    C
C             moment.                                                  C
C                                                                      C
C     AXIALIR4 - Pedro J. Pizarro, 4/20/93                             C
C                                                                      C
C             Added spin-locking field to IRICE.  See PJP notes of     C
C             12/29/92 for SLIRICE and 4/15/93 for spin flip           C
C             probabilities.                                           C
C                                                                      C
C     AXIALIR7 - Pedro J. Pizarro, 5/12/93                             C
C                                                                      C
C             Based on AXIALIR7.  Corrected expressions for W1, the    C
C             spin-lock Rabi frequency.  See PJP notes of 5/12/93.     C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine tsg

      implicit none

      integer narrmx,maxion,maxt1,nmax,nonoff
      parameter(narrmx=2048,maxion=1000,maxt1=40)

      real*8 q,m,u,b0,b2,v0,vinit,d,dx,hbar,z0,temp,wz,pi,pi2,
     1       wc,wp,wm,kb,p0,g,b1,w1,dw,dwi(maxion),mrres,
     2       umech(maxion),za(maxion),phiz(maxion),workr(2,narrmx),
     3       worki(2,narrmx),rmsv,kappa,gamma,
     4       vmbfac,vtable(narrmx),re,dts,dt1,sigres,ref,tb2,
     5       pp(maxion),pm(maxion),rdamp,tempr,tv,wzinit,rtrap,sw2,
     6       bbb,dvx,grad,tg,wzb2(maxion),phip(maxion),snoise,
     7       sigr(narrmx),sigi(narrmx),sigm(narrmx)
      real*4 rand
```

```
      integer i,j,npts,nions,ftrap(maxion),it1,idet,nvmb,nvtab,
     1        nt1,icount,idum,ishot,ioncnt,iont1,
     2        flips,filter,ifiltr
      integer*2 kran,ipad
      common/vars/q,m,u,b0,b2,v0,vinit,d,dx,hbar,z0,temp,wz,pi,pi2,
     1        wc,wp,wm,kb,p0,g,b1,w1,dw,dwi,mrres,umech,za,phiz,
     2        rmsv,kappa,gamma,vmbfac,vtable,re,dts,dt1,
     3        npts,nions,ftrap,it1,nt1,idet,nvmb,nvtab,sigres,ref,
     4        tb2,icount,kran,ipad,idum,ishot,ioncnt,workr,worki,
     5        pp,pm,rdamp,tempr,tv,wzinit,rtrap,sw2,
     6        bbb,dvx,grad,tg,wzb2,phip,flips,filter,ifiltr,snoise

      real*8 prob,dphi,spin,weff


C
C     Set up DPHI: this is a random phase increment used to simulate
C     a lack of synchronization between the two TSG periods; ions
C     perfectly in phase with the first TSG period may thus be out
C     of phase with the second TSG period
C
C     call random(rand)
C     dphi =  pi2*dble(dfloat(rand))
      dphi = 0.d+0

      w1 = 0.5d+0*g*u*b1/hbar

 10   do 100 i=1,nions

              if(ftrap(i).eq.0) goto 100      !Neglect detrapped ions
C
C     Set up spin
C
              call random(rand)
              spin = dfloat(nint(
     1              2.d+0*(dfloat(nint(dfloat(rand)))-0.5d+0))) !-1 or 1
C
C     Calculate new cyclotron radius after first TSG period
C
              weff = dsqrt(w1*w1+dwi(i)*dwi(i))
              pp(i) = pp(i) + (spin*u*grad/(2.d+0*m*(wp-wm)))*
     1              tg*(w1/weff)*dcos(phip(i))
C
C     Magnetic resonance: flip spin (or not)
C
C     Calculate PROBability of an MR transition: see PJP notes of 4/15/93
C
              prob = 0.5d+0*(w1*w1/(weff*weff))*
     1               (1.d+0-dcos(dwi(i)*dt1*dfloat(it1-1)))
              call random(rand)
              if(dfloat(rand) .le. prob .or. prob .eq. 1.d+0) then
                      spin = -1.d+0*spin
C                     write(6,*)'~spin flip for ion # ',i
                      flips = flips+1   !Spin flip counter
              endif
C
C     Calculate new cyclotron radius after second TSG period
C
              pp(i) = pp(i) + (spin*u*grad/(2.d+0*m*(wp-wm)))*
     1              tg*(w1/weff)*dcos(phip(i)+dphi)
C
C     Calculate new mechanical moment
C
              umech(i) = 0.5d+0*q*(wp*(pp(i)**2) + wm*(pm(i)**2))

 100  continue

      return
      end
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                       C
C    FUNCTION GASDEV - Numerical Recipes (1986), p. 203.
     C
C                                       C
C        Returns a normally distributed deviate with zero mean          C
C        and unit variance, using RAN1(IDUM) as the source of           C
C        uniform deviates.                    C
C                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      FUNCTION GASDEV(IDUM)
      DATA ISET/0/
      IF (ISET.EQ.0) THEN
1       V1=2.*RAN1(IDUM)-1.
        V2=2.*RAN1(IDUM)-1.
        R=V1**2+V2**2
        IF(R.GE.1.)GO TO 1
        FAC=SQRT(-2.*LOG(R)/R)
        GSET=V1*FAC
        GASDEV=V2*FAC
        ISET=1
      ELSE
        GASDEV=GSET
        ISET=0
      ENDIF
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                       C
C    FUNCTION RAN1 - Numerical Recipes (1986), pp. 196-197.
     C
C                                       C
C        Returns a uniform random deviate between 0.0 and 1.0.          C
C        Set IDUM to any negative value to initialize or                C
C        reinitialize the sequence.                                     C
C                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      FUNCTION RAN1(IDUM)
      DIMENSION R(97)
      PARAMETER (M1=259200,IA1=7141,IC1=54773,RM1=3.8580247E-6)
```

```
      PARAMETER (M2=134456,IA2=8121,IC2=28411,RM2=7.4373773E-6)
      PARAMETER (M3=243000,IA3=4561,IC3=51349)
      DATA IFF /0/
      IF (IDUM.LT.0.OR.IFF.EQ.0) THEN
        IFF=1
        IX1=MOD(IC1-IDUM,M1)
        IX1=MOD(IA1*IX1+IC1,M1)
        IX2=MOD(IX1,M2)
        IX1=MOD(IA1*IX1+IC1,M1)
        IX3=MOD(IX1,M3)
        DO 11 J=1,97
          IX1=MOD(IA1*IX1+IC1,M1)
          IX2=MOD(IA2*IX2+IC2,M2)
          R(J)=(FLOAT(IX1)+FLOAT(IX2)*RM2)*RM1
11      CONTINUE
        IDUM=1
      ENDIF
      IX1=MOD(IA1*IX1+IC1,M1)
      IX2=MOD(IA2*IX2+IC2,M2)
      IX3=MOD(IA3*IX3+IC3,M3)
      J=1+(97*IX3)/M3
      IF(J.GT.97.OR.J.LT.1)PAUSE
      RAN1=R(J)
      R(J)=(FLOAT(IX1)+FLOAT(IX2)*RM2)*RM1
      RETURN
      END
```

B.2 PLOTCUT3.FOR

```
      program plotcut3 !Rescales plot files
      implicit none

      integer narrmx
      parameter (narrmx=5000)

      real*4 xx,yy,xscale,yscale,xmin,xmax
      real*8 x(narrmx),y(narrmx),dx,sum
      integer i,icount,nmax
      character*72 filnam,filout

      nmax=70000


      open(unit=8,file=' ',status='old')
      open(unit=9,file=' ',status='new')

      xmin = -1.0
      xmax = 1.0

      do 100 i=1,nmax         !Read until reach specified range
             read(8,*,end=400) xx,yy
             if(xx.ge.xmin) goto 200
 100  continue

 200  icount = 1
      x(1) = xx
      y(1) = yy
      do 210 i=1,nmax         !Read until leave specified range
             read(8,*,end=220) xx,yy
             if(xx.gt.xmax) goto 220
             icount = icount + 1
             x(icount) = dble(xx)
             y(icount) = dble(yy)
 210  continue

 220  dx = dble(x(2)-x(1))
      write(6,*) '--->>Number of points within specified range = ',
     1       icount
      if(icount.lt.10) then
             write(6,*) 'ERROR!!! Integration subroutine SIMPSON ',
     1              'requires at least 10 points in range.'
             write(6,*) 'Program execution aborted.'
             goto 500
      endif

      call simpson(y,dx,icount,sum)    !Integrate
      write(6,*)' integral = ',sum
      if(sum.eq.0.d+0) then
             sum = 1.d+0
             write(6,*) '---->> integral set to 1'
      endif

      do 300 i=1,icount
C            y(i) = y(i)/sum
             write(9,*) sngl(x(i)),'    ',sngl(y(i))
 300  continue
      close(unit=9)
      goto 500

 400  write(6,*) '--->>No points within specified range...'
      goto 500

 500  close(unit=8)

      stop
      end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                     C
C   SIMPSON - Pedro J. Pizarro, 9/14/89
    C
C                                     C
C       Integrates a real list LIST with NPTS points spaced           C
C       by interval HH, via the alternative extended Simpson's         C
C       rule, as given on p. 108, Eq. 4.1.14 of Numerical              C
```

```
C       Recipes, 1987. Note: npts >= 10.
   C
C                                           C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine simpson(list,hh,npts,sum)

      parameter(nmax=5000)

      real*8 list(nmax),hh,sum
      integer npts

      if(npts.lt.10) then
              write(6,*)'    Subroutine SIMPSON requires npts >= 10'
              goto 100
      endif

      sum = 17.*list(1)/48. + 59.*list(2)/48. + 43.*list(3)/48.
     1       + 49.*list(4)/48.

      do 10 i=5,npts-4
              sum = sum + list(i)
 10   continue

      sum = sum + 49.*list(npts-3)/48. + 43.*list(npts-2)/48.
     1       + 59.*list(npts-1)/48. + 17.*list(npts)/48.

      sum = sum*hh

 100  return
      end
```

B.3 Y7.PAR

```
y7
1.6021892d-19                           q (coul)
1.6605655d-25                           m (kg)
1.4106171d-26                           u (J/T)
1.                                      b0 (T)
1000.                                   b2 (T/m**2)
0.05                                    vinit (V)
0.5                                     v0 (V)
1.099852                                d (cm)
1.0                                     z0 (cm)
298.                                    ion temp (K)
2.                                      detector temp (K)
83                                      nions
3.                                      vmbfac
200                                     nvmb
500                                     nvtab
100.                                    re (um)
2048                                    npts
0.1                                     sigres (Hz)
1.d+8                                   rdamp (ohm)
26000.                                  ref (Hz)
32                                      nt1
2500.                                   mrres (Hz)
50.                                     b1 (gauss)
0.03                                    dw (MHz)
0.1                                     tv (sec)
0.5                                     dx (cm)
0.1                                     tb2 (sec)
2.0                                     grad (T/m)
1.0                                     tg (sec)
200                                     nshots
3.                                      threshold
1.                                      snoise
1                                       ifiltr
```

B.4 Y7.OUT

```
 **********  Program AXIALIR8  **********
charge q  =   1.602189200000000E-019 coulomb
mass m  =   1.660565500000000E-025 kg
spin magnetic moment u  =   1.410617100000000E-026 joule/tesla
b0 =        1.000000000000000 tesla
b2 =     1000.000000000000000 gauss/square cm
initial trap potential =   5.000000000000000E-002 volt
v0 =   5.000000000000000E-001 volt
trap characteristic dimension d  =        1.099852000000000 cm
trap length / 2 = z0 =        1.000000000000000 cm
trap radius =        1.684843520216640 cm
axial frequency =     10050.780292339930000 Hz
ion temperature =      298.000000000000000 K
detector temperature =        2.000000000000000 K
number of ions generated =         83
ions generated with v <= mean v *        3.000000000000000
target number of velocities in velocity table =         200
number of entries in velocity table =         500
radius of ionizing electron beam =        99.999999999999990 um
number of time domain points for signal =        2048
signal resolution =    1.000000000000000E-001 Hz
signal time step =   4.882812500000000E-003 sec
signal detection resistance =   1.000000000000000E+008 ohm
signal detection damping =   6.150801752252941E-003 Hz
signal reference frequency =    26000.000000000000000 Hz
signal bandwidth =      204.800000000000000 Hz
rms noise voltage =    1.503659535932254E-006 V (total)
```

```
rms noise voltage per FFT point =   3.322649545167230E-008 V (total)
number of points in t1 for MR =          32
MR resolution =     2500.000000000000000 Hz
MR time step =   1.250000000000000E-005 sec
Spin lock field strength =       50.000000000000000 gauss
MR resonance offset = (1/2,1)*   3.000000000000000E-002 MHz
total time for Vinit to V0 sweep =   1.000000000000000E-001 sec
voltage with axial amplitude z0/2 =   2.529492851921968E-007 V (total)
change in x by ICR (ICR radius) =   5.000000000000000E-001 cm
total time for 0 to B2 sweep =   1.000000000000000E-001 sec
TSG gradient strength =        2.000000000000000 T/m
TSG excitation time =        1.000000000000000 sec
Spin lock frequency  =   106566.206817369000000 Hz
number of shots =         200
threshold (#standard deviations above mean FFT magnitude
          discarded by blanking) =        3.000000000000000
snoise =        1.000000000000000->full noise included
ifiltr =           1->signal filter ON
Mean ion velocity =             222.553622508337600 m/sec
Velocity table increment =        3.338304337625064 m/sec
Number of entries in velocity table =          486
Ions in MR loop #   1: 11077 trapped;  2335 through filter;     0 spin flips
Ions in MR loop #   2: 11105 trapped;  2287 through filter;  1224 spin flips
Ions in MR loop #   3: 11069 trapped;  2301 through filter;  1472 spin flips
Ions in MR loop #   4: 11069 trapped;  2282 through filter;  1241 spin flips
Ions in MR loop #   5: 11159 trapped;  2271 through filter;  1616 spin flips
Ions in MR loop #   6: 10990 trapped;  2230 through filter;   202 spin flips
Ions in MR loop #   7: 10992 trapped;  2243 through filter;   644 spin flips
Ions in MR loop #   8: 11139 trapped;  2293 through filter;  1680 spin flips
Ions in MR loop #   9: 11011 trapped;  2261 through filter;  1101 spin flips
Ions in MR loop #  10: 10971 trapped;  2261 through filter;  1672 spin flips
Ions in MR loop #  11: 11074 trapped;  2290 through filter;   737 spin flips
Ions in MR loop #  12: 11086 trapped;  2281 through filter;   193 spin flips
Ions in MR loop #  13: 11024 trapped;  2322 through filter;  1647 spin flips
Ions in MR loop #  14: 11057 trapped;  2237 through filter;  1208 spin flips
Ions in MR loop #  15: 10959 trapped;  2235 through filter;  1475 spin flips
Ions in MR loop #  16: 11068 trapped;  2280 through filter;  1220 spin flips
Ions in MR loop #  17: 11041 trapped;  2313 through filter;     0 spin flips
Ions in MR loop #  18: 11000 trapped;  2343 through filter;  1285 spin flips
Ions in MR loop #  19: 10953 trapped;  2305 through filter;  1534 spin flips
Ions in MR loop #  20: 10959 trapped;  2244 through filter;  1205 spin flips
Ions in MR loop #  21: 11123 trapped;  2346 through filter;  1672 spin flips
Ions in MR loop #  22: 11068 trapped;  2265 through filter;   181 spin flips
Ions in MR loop #  23: 11011 trapped;  2243 through filter;   671 spin flips
Ions in MR loop #  24: 11070 trapped;  2310 through filter;  1675 spin flips
Ions in MR loop #  25: 10961 trapped;  2187 through filter;  1037 spin flips
Ions in MR loop #  26: 11043 trapped;  2313 through filter;  1737 spin flips
Ions in MR loop #  27: 11061 trapped;  2201 through filter;   662 spin flips
Ions in MR loop #  28: 11113 trapped;  2336 through filter;   210 spin flips
Ions in MR loop #  29: 11145 trapped;  2293 through filter;  1633 spin flips
Ions in MR loop #  30: 11085 trapped;  2279 through filter;  1247 spin flips
Ions in MR loop #  31: 11052 trapped;  2332 through filter;  1532 spin flips
Ions in MR loop #  32: 11022 trapped;  2257 through filter;  1242 spin flips

--->>  Data saved in these formatted files:
Mag[FFT(blanked time-reversed product)] in y7m**fm.dat
```

```
Program started at:  5-13-1993  3:41:28.94
Program ended at:    5-13-1993 13:58:20.16
```

B.5 Y7.BAT

```
plotcut3 y7m01fm.dat y7m01fm.cut
plotcut3 y7m02fm.dat y7m02fm.cut
plotcut3 y7m03fm.dat y7m03fm.cut
plotcut3 y7m04fm.dat y7m04fm.cut
plotcut3 y7m05fm.dat y7m05fm.cut
plotcut3 y7m06fm.dat y7m06fm.cut
plotcut3 y7m07fm.dat y7m07fm.cut
plotcut3 y7m08fm.dat y7m08fm.cut
plotcut3 y7m09fm.dat y7m09fm.cut
plotcut3 y7m10fm.dat y7m10fm.cut
plotcut3 y7m11fm.dat y7m11fm.cut
plotcut3 y7m12fm.dat y7m12fm.cut
plotcut3 y7m13fm.dat y7m13fm.cut
plotcut3 y7m14fm.dat y7m14fm.cut
plotcut3 y7m15fm.dat y7m15fm.cut
plotcut3 y7m16fm.dat y7m16fm.cut
plotcut3 y7m17fm.dat y7m17fm.cut
plotcut3 y7m18fm.dat y7m18fm.cut
plotcut3 y7m19fm.dat y7m19fm.cut
plotcut3 y7m20fm.dat y7m20fm.cut
plotcut3 y7m21fm.dat y7m21fm.cut
plotcut3 y7m22fm.dat y7m22fm.cut
plotcut3 y7m23fm.dat y7m23fm.cut
plotcut3 y7m24fm.dat y7m24fm.cut
plotcut3 y7m25fm.dat y7m25fm.cut
plotcut3 y7m26fm.dat y7m26fm.cut
plotcut3 y7m27fm.dat y7m27fm.cut
plotcut3 y7m28fm.dat y7m28fm.cut
plotcut3 y7m29fm.dat y7m29fm.cut
plotcut3 y7m30fm.dat y7m30fm.cut
plotcut3 y7m31fm.dat y7m31fm.cut
plotcut3 y7m32fm.dat y7m32fm.cut
```

B.6 Y7.SUM

```
(******* TICK FUNCTIONS - John Marohn, 12 August 1992 *********)

dim1min = 0.0 ; dim1max = 1.0 ; ndim1units = 5 ;
dim2min = 0.0 ; dim2max = 1.0 ; ndim2units = 5 ;
dim3min = 0.0 ; dim3max = 1.0 ; ndim3units = 5 ;
zmin= 0.0 ; zmax = 1.0 ; nzunits = 5 ;

dim1units[min_,max_] := Table[ {min+(i-dim1min)(max-min)/(dim1max-dim1min),
                        ToString[N[i,3]]},
                      {i,dim1min,dim1max,(dim1max-dim1min)/(ndim1units-1)} ] ;

dim2units[min_,max_] := Table[ {min+(i-dim2min)(max-min)/(dim2max-dim2min),
                        ToString[N[i,3]]},
                      {i,dim2min,dim2max,(dim2max-dim2min)/(ndim2units-1)} ] ;
dim3units[min_,max_] := Table[ {min+(i-dim3min)(max-min)/(dim3max-dim3min),
                        ToString[N[i,3]]},
                      {i,dim3min,dim3max,(dim3max-dim3min)/(ndim3units-1)} ] ;
zunits[min_,max_] := Table[ {min+(i-zmin)(max-min)/(zmax-zmin),
                        ToString[N[i,3]]},
                      {i,zmin,zmax,(zmax-zmin)/(nzunits-1)} ] ;

(*************** End of TICK FUNCTIONS *****************)

y7m01fm = ReadList["y7m01fm.cut",{Number,Number}];
y7m02fm = ReadList["y7m02fm.cut",{Number,Number}];
y7m03fm = ReadList["y7m03fm.cut",{Number,Number}];
y7m04fm = ReadList["y7m04fm.cut",{Number,Number}];
y7m05fm = ReadList["y7m05fm.cut",{Number,Number}];
y7m06fm = ReadList["y7m06fm.cut",{Number,Number}];
y7m07fm = ReadList["y7m07fm.cut",{Number,Number}];
y7m08fm = ReadList["y7m08fm.cut",{Number,Number}];
y7m09fm = ReadList["y7m09fm.cut",{Number,Number}];
y7m10fm = ReadList["y7m10fm.cut",{Number,Number}];
y7m11fm = ReadList["y7m11fm.cut",{Number,Number}];
y7m12fm = ReadList["y7m12fm.cut",{Number,Number}];
y7m13fm = ReadList["y7m13fm.cut",{Number,Number}];
y7m14fm = ReadList["y7m14fm.cut",{Number,Number}];
y7m15fm = ReadList["y7m15fm.cut",{Number,Number}];
y7m16fm = ReadList["y7m16fm.cut",{Number,Number}];
y7m17fm = ReadList["y7m17fm.cut",{Number,Number}];
y7m18fm = ReadList["y7m18fm.cut",{Number,Number}];
y7m19fm = ReadList["y7m19fm.cut",{Number,Number}];
y7m20fm = ReadList["y7m20fm.cut",{Number,Number}];
y7m21fm = ReadList["y7m21fm.cut",{Number,Number}];
y7m22fm = ReadList["y7m22fm.cut",{Number,Number}];
y7m23fm = ReadList["y7m23fm.cut",{Number,Number}];
y7m24fm = ReadList["y7m24fm.cut",{Number,Number}];
y7m25fm = ReadList["y7m25fm.cut",{Number,Number}];
y7m26fm = ReadList["y7m26fm.cut",{Number,Number}];
y7m27fm = ReadList["y7m27fm.cut",{Number,Number}];
y7m28fm = ReadList["y7m28fm.cut",{Number,Number}];
y7m29fm = ReadList["y7m29fm.cut",{Number,Number}];
y7m30fm = ReadList["y7m30fm.cut",{Number,Number}];
y7m31fm = ReadList["y7m31fm.cut",{Number,Number}];
y7m32fm = ReadList["y7m32fm.cut",{Number,Number}];
am01fm = Table[y7m01fm[[i,2]],{i,21}]
am02fm = Table[y7m02fm[[i,2]],{i,21}]
am03fm = Table[y7m03fm[[i,2]],{i,21}]
am04fm = Table[y7m04fm[[i,2]],{i,21}]
am05fm = Table[y7m05fm[[i,2]],{i,21}]
am06fm = Table[y7m06fm[[i,2]],{i,21}]
am07fm = Table[y7m07fm[[i,2]],{i,21}]
am08fm = Table[y7m08fm[[i,2]],{i,21}]
am09fm = Table[y7m09fm[[i,2]],{i,21}]
am10fm = Table[y7m10fm[[i,2]],{i,21}]
am11fm = Table[y7m11fm[[i,2]],{i,21}]
am12fm = Table[y7m12fm[[i,2]],{i,21}]
am13fm = Table[y7m13fm[[i,2]],{i,21}]
am14fm = Table[y7m14fm[[i,2]],{i,21}]
am15fm = Table[y7m15fm[[i,2]],{i,21}]
am16fm = Table[y7m16fm[[i,2]],{i,21}]
am17fm = Table[y7m17fm[[i,2]],{i,21}]
am18fm = Table[y7m18fm[[i,2]],{i,21}]
am19fm = Table[y7m19fm[[i,2]],{i,21}]
am20fm = Table[y7m20fm[[i,2]],{i,21}]
am21fm = Table[y7m21fm[[i,2]],{i,21}]
am22fm = Table[y7m22fm[[i,2]],{i,21}]
am23fm = Table[y7m23fm[[i,2]],{i,21}]
am24fm = Table[y7m24fm[[i,2]],{i,21}]
am25fm = Table[y7m25fm[[i,2]],{i,21}]
am26fm = Table[y7m26fm[[i,2]],{i,21}]
am27fm = Table[y7m27fm[[i,2]],{i,21}]
am28fm = Table[y7m28fm[[i,2]],{i,21}]
am29fm = Table[y7m29fm[[i,2]],{i,21}]
am30fm = Table[y7m30fm[[i,2]],{i,21}]
am31fm = Table[y7m31fm[[i,2]],{i,21}]
am32fm = Table[y7m32fm[[i,2]],{i,21}]
am01fm = am01fm - Min[am01fm]
am02fm = am02fm - Min[am02fm]
am03fm = am03fm - Min[am03fm]
am04fm = am04fm - Min[am04fm]
am05fm = am05fm - Min[am05fm]
am06fm = am06fm - Min[am06fm]
am07fm = am07fm - Min[am07fm]
am08fm = am08fm - Min[am08fm]
am09fm = am09fm - Min[am09fm]
am10fm = am10fm - Min[am10fm]
am11fm = am11fm - Min[am11fm]
am12fm = am12fm - Min[am12fm]
am13fm = am13fm - Min[am13fm]
am14fm = am14fm - Min[am14fm]
am15fm = am15fm - Min[am15fm]
am16fm = am16fm - Min[am16fm]
am17fm = am17fm - Min[am17fm]
am18fm = am18fm - Min[am18fm]
am19fm = am19fm - Min[am19fm]
```

```
am20fm = am20fm - Min[am20fm]
am21fm = am21fm - Min[am21fm]
am22fm = am22fm - Min[am22fm]
am23fm = am23fm - Min[am23fm]
am24fm = am24fm - Min[am24fm]
am25fm = am25fm - Min[am25fm]
am26fm = am26fm - Min[am26fm]
am27fm = am27fm - Min[am27fm]
am28fm = am28fm - Min[am28fm]
am29fm = am29fm - Min[am29fm]
am30fm = am30fm - Min[am30fm]
am31fm = am31fm - Min[am31fm]
am32fm = am32fm - Min[am32fm]
norm01 = 0.1*(0.5*am01fm[[1]]+Sum[am01fm[[i]],{i,2,15}]+0.5*am01fm[[16]])
norm02 = 0.1*(0.5*am02fm[[1]]+Sum[am02fm[[i]],{i,2,15}]+0.5*am02fm[[16]])
norm03 = 0.1*(0.5*am03fm[[1]]+Sum[am03fm[[i]],{i,2,15}]+0.5*am03fm[[16]])
norm04 = 0.1*(0.5*am04fm[[1]]+Sum[am04fm[[i]],{i,2,15}]+0.5*am04fm[[16]])
norm05 = 0.1*(0.5*am05fm[[1]]+Sum[am05fm[[i]],{i,2,15}]+0.5*am05fm[[16]])
norm06 = 0.1*(0.5*am06fm[[1]]+Sum[am06fm[[i]],{i,2,15}]+0.5*am06fm[[16]])
norm07 = 0.1*(0.5*am07fm[[1]]+Sum[am07fm[[i]],{i,2,15}]+0.5*am07fm[[16]])
norm08 = 0.1*(0.5*am08fm[[1]]+Sum[am08fm[[i]],{i,2,15}]+0.5*am08fm[[16]])
norm09 = 0.1*(0.5*am09fm[[1]]+Sum[am09fm[[i]],{i,2,15}]+0.5*am09fm[[16]])
norm10 = 0.1*(0.5*am10fm[[1]]+Sum[am10fm[[i]],{i,2,15}]+0.5*am10fm[[16]])
norm11 = 0.1*(0.5*am11fm[[1]]+Sum[am11fm[[i]],{i,2,15}]+0.5*am11fm[[16]])
norm12 = 0.1*(0.5*am12fm[[1]]+Sum[am12fm[[i]],{i,2,15}]+0.5*am12fm[[16]])
norm13 = 0.1*(0.5*am13fm[[1]]+Sum[am13fm[[i]],{i,2,15}]+0.5*am13fm[[16]])
norm14 = 0.1*(0.5*am14fm[[1]]+Sum[am14fm[[i]],{i,2,15}]+0.5*am14fm[[16]])
norm15 = 0.1*(0.5*am15fm[[1]]+Sum[am15fm[[i]],{i,2,15}]+0.5*am15fm[[16]])
norm16 = 0.1*(0.5*am16fm[[1]]+Sum[am16fm[[i]],{i,2,15}]+0.5*am16fm[[16]])
norm17 = 0.1*(0.5*am17fm[[1]]+Sum[am17fm[[i]],{i,2,15}]+0.5*am17fm[[16]])
norm18 = 0.1*(0.5*am18fm[[1]]+Sum[am18fm[[i]],{i,2,15}]+0.5*am18fm[[16]])
norm19 = 0.1*(0.5*am19fm[[1]]+Sum[am19fm[[i]],{i,2,15}]+0.5*am19fm[[16]])
norm20 = 0.1*(0.5*am20fm[[1]]+Sum[am20fm[[i]],{i,2,15}]+0.5*am20fm[[16]])
norm21 = 0.1*(0.5*am21fm[[1]]+Sum[am21fm[[i]],{i,2,15}]+0.5*am21fm[[16]])
norm22 = 0.1*(0.5*am22fm[[1]]+Sum[am22fm[[i]],{i,2,15}]+0.5*am22fm[[16]])
norm23 = 0.1*(0.5*am23fm[[1]]+Sum[am23fm[[i]],{i,2,15}]+0.5*am23fm[[16]])
norm24 = 0.1*(0.5*am24fm[[1]]+Sum[am24fm[[i]],{i,2,15}]+0.5*am24fm[[16]])
norm25 = 0.1*(0.5*am25fm[[1]]+Sum[am25fm[[i]],{i,2,15}]+0.5*am25fm[[16]])
norm26 = 0.1*(0.5*am26fm[[1]]+Sum[am26fm[[i]],{i,2,15}]+0.5*am26fm[[16]])
norm27 = 0.1*(0.5*am27fm[[1]]+Sum[am27fm[[i]],{i,2,15}]+0.5*am27fm[[16]])
norm28 = 0.1*(0.5*am28fm[[1]]+Sum[am28fm[[i]],{i,2,15}]+0.5*am28fm[[16]])
norm29 = 0.1*(0.5*am29fm[[1]]+Sum[am29fm[[i]],{i,2,15}]+0.5*am29fm[[16]])
norm30 = 0.1*(0.5*am30fm[[1]]+Sum[am30fm[[i]],{i,2,15}]+0.5*am30fm[[16]])
norm31 = 0.1*(0.5*am31fm[[1]]+Sum[am31fm[[i]],{i,2,15}]+0.5*am31fm[[16]])
norm32 = 0.1*(0.5*am32fm[[1]]+Sum[am32fm[[i]],{i,2,15}]+0.5*am32fm[[16]])
am01fm = am01fm/norm01
am02fm = am02fm/norm02
am03fm = am03fm/norm03
am04fm = am04fm/norm04
am05fm = am05fm/norm05
am06fm = am06fm/norm06
am07fm = am07fm/norm07
am08fm = am08fm/norm08
am09fm = am09fm/norm09
am10fm = am10fm/norm10
am11fm = am11fm/norm11
am12fm = am12fm/norm12
am13fm = am13fm/norm13
am14fm = am14fm/norm14
am15fm = am15fm/norm15
am16fm = am16fm/norm16
am17fm = am17fm/norm17
am18fm = am18fm/norm18
am19fm = am19fm/norm19
am20fm = am20fm/norm20
am21fm = am21fm/norm21
am22fm = am22fm/norm22
am23fm = am23fm/norm23
am24fm = am24fm/norm24
am25fm = am25fm/norm25
am26fm = am26fm/norm26
am27fm = am27fm/norm27
am28fm = am28fm/norm28
am29fm = am29fm/norm29
am30fm = am30fm/norm30
am31fm = am31fm/norm31
am32fm = am32fm/norm32
integ01 =   0.1*( am01fm[[2]]/2  + Sum[am01fm[[i]],{i,3,9}] - am01fm[[11]] \
              + am01fm[[20]]/2 + Sum[am01fm[[i]],{i,13,19}] )
integ02 =   0.1*( am02fm[[2]]/2  + Sum[am02fm[[i]],{i,3,9}] - am02fm[[11]] \
              + am02fm[[20]]/2 + Sum[am02fm[[i]],{i,13,19}] )
integ03 =   0.1*( am03fm[[2]]/2  + Sum[am03fm[[i]],{i,3,9}] - am03fm[[11]] \
              + am03fm[[20]]/2 + Sum[am03fm[[i]],{i,13,19}] )
integ04 =   0.1*( am04fm[[2]]/2  + Sum[am04fm[[i]],{i,3,9}] - am04fm[[11]] \
              + am04fm[[20]]/2 + Sum[am04fm[[i]],{i,13,19}] )
integ05 =   0.1*( am05fm[[2]]/2  + Sum[am05fm[[i]],{i,3,9}] - am05fm[[11]] \
              + am05fm[[20]]/2 + Sum[am05fm[[i]],{i,13,19}] )
integ06 =   0.1*( am06fm[[2]]/2  + Sum[am06fm[[i]],{i,3,9}] - am06fm[[11]] \
              + am06fm[[20]]/2 + Sum[am06fm[[i]],{i,13,19}] )
integ07 =   0.1*( am07fm[[2]]/2  + Sum[am07fm[[i]],{i,3,9}] - am07fm[[11]] \
              + am07fm[[20]]/2 + Sum[am07fm[[i]],{i,13,19}] )
integ08 =   0.1*( am08fm[[2]]/2  + Sum[am08fm[[i]],{i,3,9}] - am08fm[[11]] \
              + am08fm[[20]]/2 + Sum[am08fm[[i]],{i,13,19}] )
integ09 =   0.1*( am09fm[[2]]/2  + Sum[am09fm[[i]],{i,3,9}] - am09fm[[11]] \
              + am09fm[[20]]/2 + Sum[am09fm[[i]],{i,13,19}] )
integ10 =   0.1*( am10fm[[2]]/2  + Sum[am10fm[[i]],{i,3,9}] - am10fm[[11]] \
              + am10fm[[20]]/2 + Sum[am10fm[[i]],{i,13,19}] )
integ11 =   0.1*( am11fm[[2]]/2  + Sum[am11fm[[i]],{i,3,9}] - am11fm[[11]] \
              + am11fm[[20]]/2 + Sum[am11fm[[i]],{i,13,19}] )
integ12 =   0.1*( am12fm[[2]]/2  + Sum[am12fm[[i]],{i,3,9}] - am12fm[[11]] \
              + am12fm[[20]]/2 + Sum[am12fm[[i]],{i,13,19}] )
integ13 =   0.1*( am13fm[[2]]/2  + Sum[am13fm[[i]],{i,3,9}] - am13fm[[11]] \
              + am13fm[[20]]/2 + Sum[am13fm[[i]],{i,13,19}] )
integ14 =   0.1*( am14fm[[2]]/2  + Sum[am14fm[[i]],{i,3,9}] - am14fm[[11]] \
              + am14fm[[20]]/2 + Sum[am14fm[[i]],{i,13,19}] )
integ15 =   0.1*( am15fm[[2]]/2  + Sum[am15fm[[i]],{i,3,9}] - am15fm[[11]] \
              + am15fm[[20]]/2 + Sum[am15fm[[i]],{i,13,19}] )
integ16 =   0.1*( am16fm[[2]]/2  + Sum[am16fm[[i]],{i,3,9}] - am16fm[[11]] \
              + am16fm[[20]]/2 + Sum[am16fm[[i]],{i,13,19}] )
integ17 =   0.1*( am17fm[[2]]/2  + Sum[am17fm[[i]],{i,3,9}] - am17fm[[11]] \
              + am17fm[[20]]/2 + Sum[am17fm[[i]],{i,13,19}] )
integ18 =   0.1*( am18fm[[2]]/2  + Sum[am18fm[[i]],{i,3,9}] - am18fm[[11]] \
```

```
              + am18fm[[20]]/2 + Sum[am18fm[[i]],{i,13,19}] )
integ19 =   0.1*( am19fm[[2]]/2  + Sum[am19fm[[i]],{i,3,9}] - am19fm[[11]] \
              + am19fm[[20]]/2 + Sum[am19fm[[i]],{i,13,19}] )
integ20 =   0.1*( am20fm[[2]]/2  + Sum[am20fm[[i]],{i,3,9}] - am20fm[[11]] \
              + am20fm[[20]]/2 + Sum[am20fm[[i]],{i,13,19}] )
integ21 =   0.1*( am21fm[[2]]/2  + Sum[am21fm[[i]],{i,3,9}] - am21fm[[11]] \
              + am21fm[[20]]/2 + Sum[am21fm[[i]],{i,13,19}] )
integ22 =   0.1*( am22fm[[2]]/2  + Sum[am22fm[[i]],{i,3,9}] - am22fm[[11]] \
              + am22fm[[20]]/2 + Sum[am22fm[[i]],{i,13,19}] )
integ23 =   0.1*( am23fm[[2]]/2  + Sum[am23fm[[i]],{i,3,9}] - am23fm[[11]] \
              + am23fm[[20]]/2 + Sum[am23fm[[i]],{i,13,19}] )
integ24 =   0.1*( am24fm[[2]]/2  + Sum[am24fm[[i]],{i,3,9}] - am24fm[[11]] \
              + am24fm[[20]]/2 + Sum[am24fm[[i]],{i,13,19}] )
integ25 =   0.1*( am25fm[[2]]/2  + Sum[am25fm[[i]],{i,3,9}] - am25fm[[11]] \
              + am25fm[[20]]/2 + Sum[am25fm[[i]],{i,13,19}] )
integ26 =   0.1*( am26fm[[2]]/2  + Sum[am26fm[[i]],{i,3,9}] - am26fm[[11]] \
              + am26fm[[20]]/2 + Sum[am26fm[[i]],{i,13,19}] )
integ27 =   0.1*( am27fm[[2]]/2  + Sum[am27fm[[i]],{i,3,9}] - am27fm[[11]] \
              + am27fm[[20]]/2 + Sum[am27fm[[i]],{i,13,19}] )
integ28 =   0.1*( am28fm[[2]]/2  + Sum[am28fm[[i]],{i,3,9}] - am28fm[[11]] \
              + am28fm[[20]]/2 + Sum[am28fm[[i]],{i,13,19}] )
integ29 =   0.1*( am29fm[[2]]/2  + Sum[am29fm[[i]],{i,3,9}] - am29fm[[11]] \
              + am29fm[[20]]/2 + Sum[am29fm[[i]],{i,13,19}] )
integ30 =   0.1*( am30fm[[2]]/2  + Sum[am30fm[[i]],{i,3,9}] - am30fm[[11]] \
              + am30fm[[20]]/2 + Sum[am30fm[[i]],{i,13,19}] )
integ31 =   0.1*( am31fm[[2]]/2  + Sum[am31fm[[i]],{i,3,9}] - am31fm[[11]] \
              + am31fm[[20]]/2 + Sum[am31fm[[i]],{i,13,19}] )
integ32 =   0.1*( am32fm[[2]]/2  + Sum[am32fm[[i]],{i,3,9}] - am32fm[[11]] \
              + am32fm[[20]]/2 + Sum[am32fm[[i]],{i,13,19}] )
integ = {integ01,integ02,integ03,integ04,integ05,integ06,integ07,integ08, \
         integ09,integ10,integ11,integ12,integ13,integ14,integ15,integ16, \
         integ17,integ18,integ19,integ20,integ21,integ22,integ23,integ24, \
         integ25,integ26,integ27,integ28,integ29,integ30,integ31,integ32}
finteg = Fourier[integ]
finteg2 = Join[Table[finteg[[i]],{i,17,32}],Table[finteg[[i]],{i,1,17}]]

dim1min = -1. ; dim1max = 1. ; ndim1units = 5 ;
dim2min = 0. ; dim2max = 15*12.5 ; ndim2units = 5 ;
dim3min = -40. ; dim3max = 40. ; ndim3units = 5 ;
ga1 = ListPlot3D[{am01fm,am02fm,am03fm,am04fm,am05fm,am06fm,am07fm,am08fm,\
                  am09fm,am10fm,am11fm,am12fm,am13fm,am14fm,am15fm,am16fm,\
                  am17fm,am18fm,am19fm,am20fm,am21fm,am22fm,am23fm,am24fm,\
                  am25fm,am26fm,am27fm,am28fm,am29fm,am30fm,am31fm,am32fm},\
        PlotRange->All,Ticks->{dim1units,dim2units,None},Boxed->False,\
        Axes->{True,True,False},AxesLabel->{"Frequency (Hz)","t1 (usec)"}]
ga2 = ListPlot[integ,PlotJoined->True,PlotRange->All,Frame->True,\
        FrameLabel->{"t1 (usec)",None,"Sidebands - Central Peak",None},\
        FrameTicks->{dim2units,Automatic},Axes->False]
ga3 = ListPlot[Abs[finteg2],PlotJoined->True,PlotRange->All,Frame->True, \
        FrameLabel->{"Frequency (kHz)",None},\
        FrameTicks->{dim3units,Automatic},Axes->False]
ga = GraphicsArray[{{ga1},{ga2},{ga3}}, \
                        PlotLabel->"y7m**fm.cut, y7.sum, 5/13/93"];
Show[ga]
```